AFRL-RI-RS-TR-2018-036

STREAM PROCESSING ALGORITHMS FOR DYNAMIC 3D SCENE ANALYSIS

UNIVERSITY OF MISSOURI

FEBRUARY 2018

FINAL TECHNICAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■ **UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09. This report is available to the general public, including foreign nations. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RI-RS-TR-2018-036 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE CHIEF ENGINEER:

/ S /
DAVID D. FERRIS
Work Unit Manager

/ S /
JON S. JONES
Technical Advisor, Information Intelligence
Systems and Analysis Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
FEB 2018	FINAL TECHNICAL REPORT	JAN 2014 – AUG 2017

4. TITLE AND SUBTITLE

STREAM PROCESSING ALGORITHMS FOR DYNAMIC 3D SCENE ANALYSIS

5a. CONTRACT NUMBER
FA8750-14-2-0072

5b. GRANT NUMBER
N/A

5c. PROGRAM ELEMENT NUMBER
62788F

6. AUTHOR(S)

Kannappan Palaniappan

5d. PROJECT NUMBER
VMRG

5e. TASK NUMBER
IE

5f. WORK UNIT NUMBER
D1

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

University of Missouri
Department of Electrical Engineering and Computer Science
310 Jesse Hall
Columbia MO 65211-1230

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory/RIED
525 Brooks Road
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RI

11. SPONSOR/MONITOR'S REPORT NUMBER
AFRL-RI-RS-TR-2018-036

12. DISTRIBUTION AVAILABILITY STATEMENT

Approved for Public Release; Distribution Unlimited. This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The goal of this effort was to take the computation closer to the sensor when it made sense to do so. Wide Area Motion Imagery (WAMI) data was used given the high bandwidth requirements associated with WAMI. Our focus was on; feature detection and matching techniques along the video sequence; developing efficient structure-from-motion and bundle adjustment techniques for high resolution aerial imagery; designing 3D reconstruction algorithm to run on metropolitan-scale data; analytical image stabilization and geo-registration; and developing moving object detection and tracking algorithms. The creation of 3D models from airborne wide area motion imagery (WAMI) will facilitate a number of capabilities including improved geo-registration, increased reliability of target tracking through occlusions and shadows, onboard processing for video object analysis, better modeling of object motion dynamics, and enable high compression of large scale scene imagery for efficient real time downlinks.

15. SUBJECT TERMS

3D Reconstruction, Target Detection and Tracking, Geo-registration, onboard processing, structure-from-motion, feature detection, bundle adjustment

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
					DAVID D. FERRIS
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER (*Include area code)*
U	U	U	UU	99	

Contents

List of Figures

List of Tables

1 Summary

The goal on this effort, as a part of the AFRL ECP, was to take the computation closer to the sensor. Our focus was on feature detection and matching techniques along the video sequence, developing efficient structure-from-motion and bundle adjustment techniques for high resolution aerial imagery, designing 3D reconstruction algorithm to run on metropolitan-scale data, analytical image stabilization and georegistration, and developing moving object detection and tracking algorithms.

Processing close to the sensor is becoming essential given the large image sensor data volumes for maximum endurance, coordinated persistent surveillance within contested environments and intelligent decision making using autonomous platforms. The creation of 3D models from airborne wide area motion imagery (WAMI) will facilitate a number of capabilities including improved geo-registration, increased reliability of target tracking through occlusions and shadows, onboard processing for video object analysis, better modeling of object motion dynamics, and enable high compression of large scale scene imagery for efficient real time downlinks. This ECP accomplished the previous work on creating a robust 3D capability using new scalable multiview reconstruction algorithms. Scalability to large regions and higher resolution point clouds, improved accuracy of structures and parallelization for a factor of ten reduction in computational speed were the focus of this ECP effort, which has been successfully achieved. Automatic moving object detection and segmentation is one of the fundamental low-level tasks for many of the WAMI surveillance systems including traffic monitoring, activity and behavior recognition and tracking applications. We developed an automatic moving vehicle detection system for wide area aerial imagery based on semantic fusion of flux trace and building mask information. An average precision of 90% and recall of 80% have been reported for 200, $2k \times 2k$ cropped region of interest from Albuquerque urban imagery.

This effort fulfilled the proposed three directions of research defined in the ECP extension within scope of the original proposal: a) Improved accuracy of multiview 3D reconstruction of urban scenes using a volumetric multipass voting algorithm that uses bundle adjustment to ensure accurate camera pose and position information, uniformly dense feature extraction for sufficient vote accumulation, appropriate temporal sampling, point cloud extraction from the vote volume, vertex coloring and post processing to estimate accurate surface geometries. b) Stream processing to enable integration of video analysis algorithms developed for object motion analysis under the CETE project. c) Parallelization of components of the visual exploitation algorithms including video motion analysis, 3D reconstruction. The flow of processing for the 3D reconstruction algorithm is shown in the system diagram below with the major modules identified. Our pipeline in this effort was a novel approach to recovering 3D structure of urban scenes with dense buildings using a 3D multipass voxel voting method. Some intermediate stages of the pipeline including sensor measurements, bundle adjustment, voting, post processing and point cloud coloring and rendering are shown below.

2 Introduction

Processing close to the sensor is becoming essential given the large image sensor data volumes for maximum endurance, coordinated persistent surveillance within contested environments and intelligent decision making using autonomous platforms. The creation of 3D models from airborne wide area motion imagery (WAMI) will facilitate a number of capabilities including improved geo-registration, increased reliability of target tracking through occlusions and shadows, onboard processing for video object analysis, better modeling of object motion dynamics, and enable high compression of large scale scene imagery for efficient real time downlinks. This technical report presents our endeavors within the scope of this proposal on creating a robust 3D capability using new scalable multiview reconstruction algorithms. Parallelized visual processing algorithms for heterogeneous multi-core architectures will provide a path to achieve real time requirements for the computationally intensive 3D reconstruction tasks. Under the current project we have successfully demonstrated 3D reconstruction of several urban areas including Albuquerque, Los Angeles, Berkeley, Columbia, Coit Tower/San Francisco and Four Hills. We have also shown a great achievement in improving the accuracy of camera metadata using a fast and robust structure

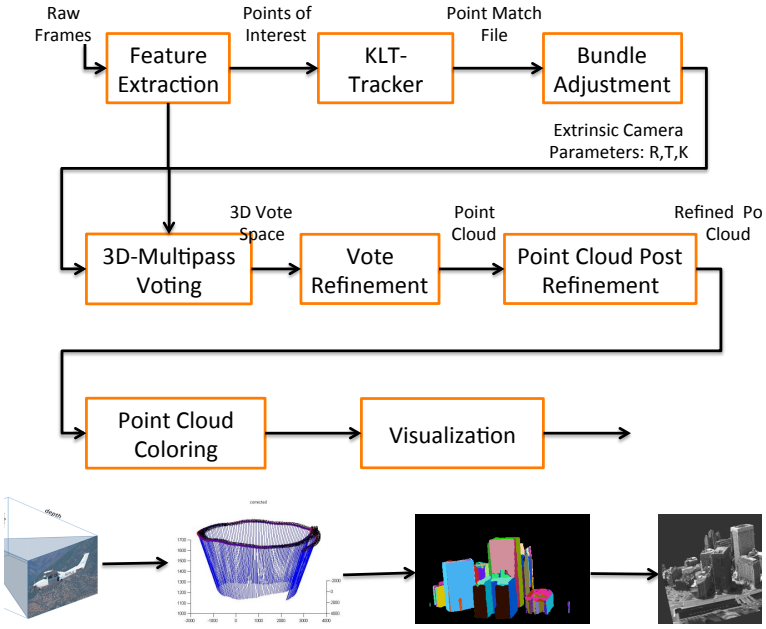


Figure 1: The 3D processing pipeline flowchart showing key modules.

from motion and bundle adjustment algorithm. By fusion of depth masks of the scene obtained from 3D reconstruction and a flux tensor based detection method, we have drastically improved our multi target tracking algorithm. Below is the list of our deliverable on this effort.

1. Bundle adjustment algorithm to improve noisy camera metadata that exploits the availability of initial metadata from on-board sensors, WAMI flightpath geometry to ensure that rays across multiple views for the same feature will converge appropriately using co-visibility constraints. (ACCOMPLISHED)
2. Multi-pass voxel-based voting algorithm with a tradeoff between the size of the voxel space and the cost of computation and memory requirements for a given level of 3D reconstruction fidelity that is scalable to large scenes. (ACCOMPLISHED)
3. Parallelizing critical parts of the overall 3D processing pipeline using a combination of methods including converting compute intensive functions from Matlab to C++, using multithreading, porting to parallel GPU implementations as necessary. (ACCOMPLISHED)
4. Reducing the current total time taken from about 100 hours for reconstructing the downtown LA region using a Matlab implementation to 10 hours using a mixture of Matlab, C++ and GPU CUDA. (ACCOMPLISHED)
5. Extension of implementation of multiscale structure tensor point feature extraction and persistent feature alignment for improved positional accuracy and to incorporate linear features. (ACCOMPLISHED)
6. Vote refinement algorithm to improve the quality of the point cloud by removing noisy 3D vertices or outlier voxels, fill in gaps and holes in surfaces, plane fitting and smoothly interpolate across curved surfaces. (ACCOMPLISHED)
7. Point cloud coloring algorithm using information about local spatially consistent patches in the 3D scene obtained by surface clustering and normal estimation. (ACCOMPLISHED)
8. Interactive visualization tool called Nimbus or Stratus for viewing point clouds and their geometric properties. (ACCOMPLISHED)
9. Visual motion analysis algorithms including LOFT and CSURF for transition to AFRL. (ACCOMPLISHED)
10. Many core parallel implementation of some of the computationally intensive modules such as feature extraction, bundle adjustment, 3D voting and point cloud processing. (ACCOMPLISHED)

11. Collaboration with Transparent Sky if subcontract funds are available for improved geo-registration, support for high dynamic range image acquisition and processing to deal with low contrast and shadow regions, and Bayesian super-resolution to improve target tracking. (ACCOMPLISHED)

12. Documentation for technical work completed and information gained as part of this ECP project. Provide a description of important observations, nature of problems, positive and negative results, and various design criteria established, procedures followed, or lessons learned.

13. Attend and present at technical review meetings for the overall CETE program.

14. Deliver software programs developed, assembled, or acquired to accomplish the video based 3D reconstruction algorithm.

3 Methods, Assumptions and Procedures

3.1 Bundle Adjustment (BA) for Camera Metadata

The use of Global Positioning System (GPS) and Inertial Measurement Unit (IMU) sensors to track the 3D path of platforms and cameras is becoming more widely available and is routinely used in aerial navigation and imaging [38] as well as hand held navigation devices [114, 4, 134]. The camera path and pose information is used to support robust, real-time 3D scene reconstruction using Structure-from-Motion algorithms (SfM) [104, 174, 184, 17, 14, 114, 23, 15, 18, 16] and direct georegistration. Irschara et al. in [104] observe that, "These systems rely on highly accurate geo-referencing devices that are calibrated and the delivered pose and orientation estimates are often superior to the one obtained by image based methods (i.e. subpixel accurate image registration)". However, many (inexpensive) aerial platforms produce IMU and GPS values of limited accuracy due to measurement and timing errors which then need to be refined for accurate SfM [104]. Extracting and incorporating 3D information in Wide Area Motion Imagery (WAMI) processing [160, 39] will be very useful for mitigating parallax effects in video summarization [208, 209], better stabilization and appearance models for tracking [155], depth map filtering of motion detections [175, 159], and improving video analytics like object tracking [169, 166]. In this project a fast and robust camera pose refinement and SfM method is proposed for sequential aerial imagery applicable for reliable georegistration and vehicle tracking tasks. The data flow of the pipeline is depicted in Fig. 2. The pipeline receives 2D aerial images which were sequentially acquired as well as approximate (noisy) pose information from on-board sensors that provide location (using a GPS) and orientation (using an IMU). Point of interests (features) are extracted in each image. The extracted features from each two adjacent frames within the sequence (starting from the first frame) are compared against each other. Going towards the next frames, sequentially matched features along the sequence are used to build tracks of interest points. 2D feature points in each track are triangulated (linear method [95]) using the noisy camera parameters (from metadata) to estimate a corresponding 3D point in the scene. A non-linear optimization method (Levenberg-Marquardt) is used to jointly refine the camera parameters and estimated 3D points. In contrast to other existing methods, the noisy metadata are directly used as the main source for initializing the camera poses without estimating them through (or combining them with) image-based approaches such as the five-point algorithm [152, 102, 8]. Outliers or mismatch errors can adversely affect the non-linear optimization approaches and in order to mitigate their impacts a robust function is proposed which is adaptive with respect to the persistency of each tracked feature. The proposed approach neither needs to estimate camera parameters nor to classify inliers/outliers in early stages of the pipeline. As a direct consequence, consensus combinatorial methods (RANSAC or its variations– see Fig. 3) are avoided which so far have been a core part of most SfM and georegistration pipelines to combat measurement noise, outliers and mismatch errors in point correspondences while estimating the camera parameters. Fig. 3 depicts the pipeline of a conventional SfM, while our proposed pipeline is shown in Fig. 4.

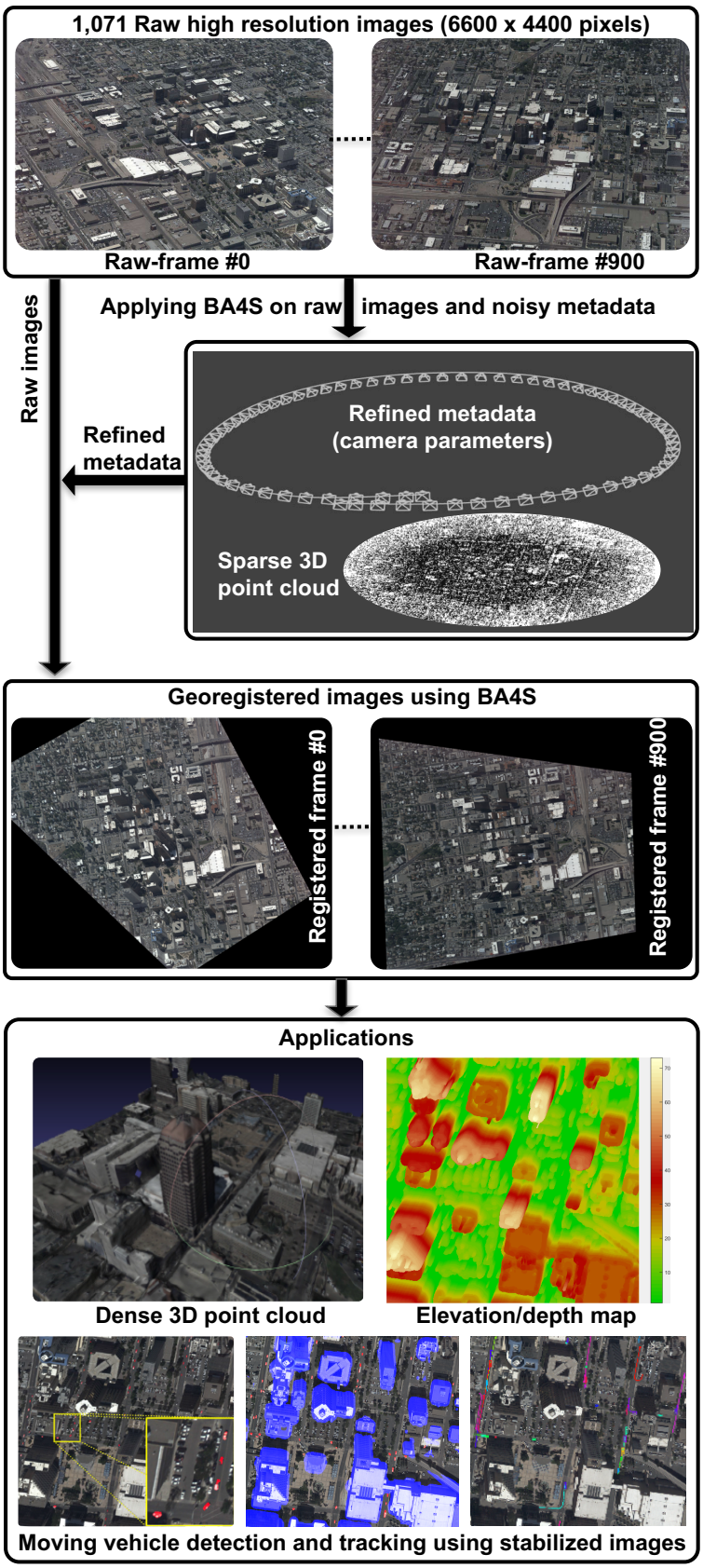


Figure 2: Overall view (data flow) of the proposed pipeline. Raw ultra high resolution images (6,600×4,400 pixels) together with noisy metadata, acquired from an airborne platform flying over Albuquerque-NM downtown, are the inputs to the pipeline. BA4S processes them in a very short amount of time (in this case less than 12 minutes for 1071 images). The BA4S's outputs are the refined camera parameters and sparse point cloud. Georegistration is performed using the proposed analytical homographies. Some applications for the BA4S are shown in the bottom box: BA4S's outputs are used to produce a dense point cloud (using PMVS[81]). The georegistered images are then used together with available 3D information (depth of the buildings from the mentioned dense point cloud) to perform moving vehicle detection and tracking[175, 159].

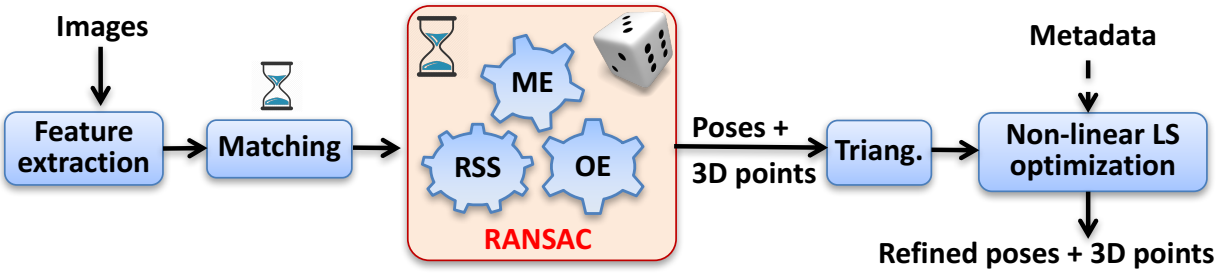


Figure 3: A conventional SfM pipeline. The acronyms RSS, ME and OE used in this figure stand for *Random Subset Selection*, *Model Estimation* and *Outlier Elimination*, respectively. In the conventional methods, iterative RANSAC (or its variations) is inevitable which includes randomness and time consumption.

3.1.1 Related Work

In the computer vision community the camera parameters (pose) are referred to as *intrinsic* and *extrinsic*, while in photogrammetry, these same metadata are known as *interior* and *exterior* parameters and the estimation process is referred to as *resectioning*. Precise estimates of these parameters are critical for practical computer vision applications (particularly dense 3D reconstruction) and accurate aerial photogrammetry. BA, considered as the gold standard for refinement of camera metadata [197, 136, 95, 223], is a classical and well studied problem in computer vision and photogrammetry dating back more than three decades [132, 203, 95]. BA uses initial estimates of camera poses to improve the metadata by minimizing reprojection errors [64, 147, 203], but is computationally expensive having a general computational complexity of $O((n+m)^3)$ and memory requirement of $O(mn(m+n))$ for n cameras and m scene 3D points [140, 79]. Conventional BA shows satisfactory convergence if sufficiently accurate initial estimates are provided either from image feature matching-based essential matrix estimation [102, 8] or combined with on-board sensor measurements [124, 78, 184]. A comprehensive introduction to BA in [203] covers a wide spectrum of methods and issues. Due to the recent surge in developing large scale 3D reconstructions using unorganized internet photos, smartphones as well as organized or sequential aerial imagery, there has been renewed interest in making BA more robust, scalable and accurate [10, 106, 118, 102, 188]. Recent methods include Sparse BA [133, 112, 63], Incremental BA [124] and Parallel BA [217, 215]. Several optimization methods for BA are compared in [106] and the conjugate gradient approach is shown to produce better results in terms of speed and convergence. A SfM method, called *Mavmap*, is proposed in [184] which leverages the temporal consistency of aerial images and availability of metadata to speed up the performance. Recently, MapTK is introduced in [122] as an open source SfM specially designed for aerial video which is able to incorporate the metadata in the pipeline. As we will present in the experiment section, MapTK is not robust enough and runs slower than our proposed pipeline with a lower accuracy, as it follows the conventional SfM pipeline. In [184], VisualSFM/Bundler [217] was considered as the most advanced and widely used system for automated and efficient 3D reconstruction from 2D images. However, as stated in [184], it is not efficient for aerial imagery and also has no integration of IMU priors. In [104], the authors used a view selection strategy to speedup SfM but had limited success using a robust BA, *"Through our experiments, robust bundle adjustment was not able to converge to a true solution from raw IMU initialized projection matrices"*. Although the authors proposed a robust BA and tested it, their approach was not sufficient to improve the camera parameters when raw metadata was used. In our work, we successfully used our proposed robust BA (BA4S) to refine inaccurate camera parameters. BA4S proposes a different SfM pipeline and camera motion model that enables inaccurate sensor measurements to be directly used as initial values for BA optimization and without any early-stage/explicit feature mismatch filtering (i.e. no RANSAC or its variants). Many approaches to use GPS and IMU measurements for refining camera parameters have been proposed, especially in the robotics community. However, GPS and IMU measurements have been used mostly as ancillary information along with other pose estimation methods through the essential matrix (e.g. Five-Point algorithm) in computer vision [124, 78] or resectioning in photogrammetry. For example, in [78, 124, 174, 41], platform and sensor GPS and IMU measurements are fused with an SfM approach using an Extended Kalman Filter or as extra constraints in

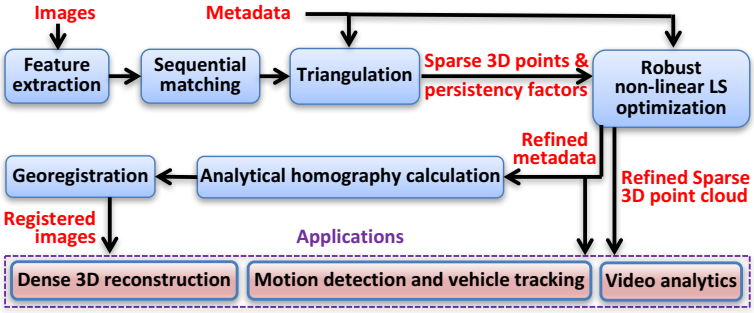


Figure 4: Developed SfM and georegistration pipeline. Noisy metadata are directly used for triangulation and robust non-linear LS optimization, while RANSAC (or its variations) is avoided. The main pipeline produces three types of outputs including: refined camera parameters, sparse 3D point cloud and georegistered/stabilized images. Homography transformations in the georegistration are directly (analytically) computed from refined camera parameters (no frame-to-frame or piecewise homography estimation). Three of the primary applications of this pipeline are shown. One can obtain a dense 3D point cloud using a dense 3D reconstruction algorithm. Moving vehicle detection can be applied on the well stabilized images for the purpose of tracking (see [175, 159] for details). Video analytics [208, 209] is another demanding application which can benefit from the proposed georegistration pipeline.

BA in order to produce 3D reconstructions and not directly as in our proposed robust BA4S approach.

3.1.2 Structure-from-Motion and Bundle Adjustment using BA4S

This section introduces details of BA4S, our developed SfM and BA algorithm. A flow diagram of the pipeline is presented in Fig. 4. We start with describing our feature extraction and matching strategy (the first three top-left blocks) and continue with introducing our developed approach for camera pose refinement to optimize the intrinsic (interior) and extrinsic (exterior) camera parameters using a persistency-based robust error function.

3.1.2.1 Features, Sequential Matching and Tracking

In sequential image capture, we know which frames are adjacent to each other, as in persistent aerial WAMI [160] or hyper-lapse first person videos [114]. By leveraging this powerful temporal consistency constraint between images as prior information, we reduce the time complexity of matching, n cameras, from $O(n^2)$ to $O(n)$, without compromising the quality of BA results [184]. In our proposed approach, interest points are extracted from each image using a robust local feature detector. Starting from the first frame, for each two successive image frames, the descriptors associated with interest points are compared with successive matches building up a set of *feature tracks* without using RANSAC. A feature track provides evidence that a potentially unique 3D point in the scene has been observed in a consecutive set of image frames. Fig. 5 illustrates a scene 3D point, $\mathbf{X}_j$, being observed by γ_j cameras, $\mathcal{C}_k \dots \mathcal{C}_{(k+\gamma_j-1)}$, in a row. Its corresponding track, τ_j, can be defined as

$$\tau_j = \{\mathbf{x}_{jk}, \mathbf{x}_{j(k+1)}, \dots \mathbf{x}_{j(k+\gamma_j-1)} \mid k \in 1 \dots n, j \in 1 \dots m, \; 2 \leq \gamma_j \leq n\} \quad (1)$$

where $\mathbf{x}_{jk}$ denotes the 2D image point of $\mathbf{X}_j$ in the image plane of $\mathcal{C}_k$, γ_j is the number of cameras sequentially observing the 3D point $\mathbf{X}_j$, and m is number of 3D points/tracks. A whole set of tracks in a dataset is presented by $\mathrm{T} = \{\tau_1, \tau_2, \dots \tau_m\}$. In our SfM approach, we consider γ_j as a persistency factor related to track τ_j. Indeed, γ_j, can be thought as a factor that measures the temporal *co-visibility* of the j-th 3D point in the image sequence. Temporal co-visibility was used in the literature for other purposes such as object recognition [88]. Here we exploit it as a robustness parameter reflecting the reliability in identifying a 3D scene point. Each track (i.e. estimated 3D feature point trajectory) has an associated persistency factor. After building all tracks, T, the population statistics of track persistency factor for m 3D points are estimated including the mean, $\mu_F = \frac{1}{m}\sum_{j=1}^{m}\gamma_j$ and standard deviation, σ_F. These first and second order track persistency statistics are used to appropriately weight each track in a novel manner in the BA optimization formulation.

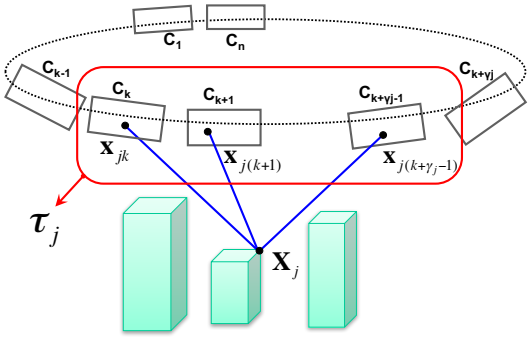


Figure 5: Illustration of sequential tracking. A scene 3D point, $\mathbf{X}_j$, is observed by γ_j cameras, $\mathcal{C}_k \ldots \mathcal{C}_{(k+\gamma_j-1)}$, in a row. They constitute a track set, τ_j, with the members defined in (1).

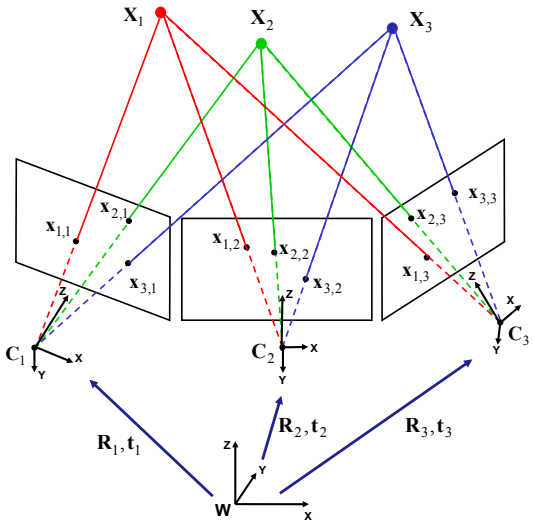


Figure 6: Bundle of rays between each camera center and the set of 3D points. This is an exemplary case in which three 3D points are observed by three cameras.

3.1.2.2 Robust Non-linear LS Optimization

Bundle Adjustment (BA) refers to the problem of jointly refining camera parameters and 3D structure in an optimal manner often using reprojection error as the quality metric. Given a set of n cameras (see Fig. 6), with arbitrary poses (translations, orientations) and, m points, BA optimization is defined as a least squares minimization using the L_2-norm or sum-of-squared reprojection errors:

$$E = \min_{\mathbf{R}_i, \mathbf{t}_i, \mathbf{K}_i, \mathbf{X}_j} \sum_{i=1}^{n} \sum_{j=1}^{m} \|\mathbf{x}_{ji} - g(\mathbf{X}_j, \mathbf{R}_i, \mathbf{t}_i, \mathbf{K}_i)\|^2 \tag{2}$$

where $\mathbf{R}_i$, $\mathbf{t}_i$, $\mathbf{K}_i$ are respectively the rotation matrix, translation vector and (intrinsic) calibration matrix of the ith camera, $\mathbf{X}_j$ is the j-th 3D point in the scene and observation $\mathbf{x}_{ji}$ is the 2D image coordinates of feature $\mathbf{X}_j$ in camera i and the L_2 norm is used. The mapping $g(\mathbf{X}_j, \mathbf{R}_i, \mathbf{t}_i, \mathbf{K}_i)$ is a transformation model which projects a 3D point $\mathbf{X}_j$ onto the image plane of camera i using its extrinsic, $\mathbf{R}_i$ and $\mathbf{t}_i$, and intrinsic parameters, $\mathbf{K}_i$, defined as:

$$g(\mathbf{X}_j, \mathbf{R}_i, \mathbf{t}_i, \mathbf{K}_i) \sim \mathbf{P}_i \mathbf{X}_{ji} \tag{3}$$

where $\mathbf{P}_i$ is the projection matrix of camera i, defined as:

$$\mathbf{P}_i = \mathbf{K}_i \left[\mathbf{R}_i | \mathbf{t}_i\right]. \tag{4}$$

The intrinsic parameters of the camera(s) are defined as:

$$\mathbf{K}_i = \begin{bmatrix} f_i & 0 & u_i \\ 0 & f_i & v_i \\ 0 & 0 & 1 \end{bmatrix}. \tag{5}$$

f_i being the focal length in pixels and the $(u_i, v_i)^T$ is the camera principal point. One can consider radial distortion parameters in the camera calibration matrix for both computing the reprojection error and also parameters to be optimized in BA (see [222, 203, 216, 68] for more details). The extrinsic matrix is composed of the rotation $\mathbf{R}_i$ and translation $\mathbf{t_i}$ of the camera:

$$[\mathbf{R_i}|\mathbf{t_i}] = \left(\begin{array}{ccc|c} r_{11,i} & r_{12,i} & r_{13,i} & t_{x,i} \\ r_{21,i} & r_{22,i} & r_{23,i} & t_{y,i} \\ r_{31,i} & r_{32,i} & r_{33,i} & t_{z,i} \end{array}\right). \tag{6}$$

However, due to errors in the metadata and feature matching outliers, $g(\mathbf{X}_j, \mathbf{R}_i, \mathbf{t}_i, \mathbf{K}_i) \neq \mathbf{x}_{ji}$, and the minimization problem seeks a statistically optimal estimate for the camera projection matrices $\mathbf{P}_i$ and 3D feature points $\mathbf{X}_j$. The L_2 minimization of the reprojection error involves adjusting the bundle of rays between each camera center and the set of 3D points which is a non-linear constrained optimization problem. Note that the above minimization is equivalent to finding a maximum likelihood solution assuming that the measurement noise is Gaussian; see [203, 95] for more details. There exist various methods to solve the non-linear least squares problem including implicit trust region and Levenberg-Marquardt methods that are well established in the BA literature [133, 106].

The selection of 2D feature point correspondences is one of the most critical steps in image-based multi-view reconstruction [13]. Feature correspondences are usually contaminated by outliers, that is matching errors or incorrect data associations. The outliers or mismatches may be caused by occlusions, repetitive patterns, illumination changes, shadows, image noise and blur for which the assumptions of the feature detector and descriptors are not satisfied [79]. On the other hand, BA which is usually solved using the Levenberg-Marquardt numerical method [203] is highly sensitive to the presence of feature correspondence outliers [13]. Mismatches can cause problems for the standard least squares approach; as stressed in [27] even a single mismatch can globally affect the result. This leads to sub-optimal parameter estimation, and in the worst case a feasible solution is not found [13, 151]. This is even more problematic in high resolution images that have a large number of features and potential correspondences which increases the probability of association or matching errors. Furthermore, aerial images have a high degree of parallax making matching and feature tracking a much more difficult problem.

Generally the mismatches are explicitly excluded from the set of potential feature correspondence in the early stages of the conventional SfM pipeline (Fig. 3) well before the BA optimization stage. In this approach the initial camera parameters are simultaneously estimated while explicitly detecting and eliminating outliers usually by applying different variations of RANSAC. In our proposed SfM approach (Fig. 4), we show that we can bypass the explicit RANSAC-based outlier elimination step by using an appropriate robust error measure.

Robust error functions also known as M-estimators are popular in robust statistics and reduce the influence of outliers in estimation problems. We have observed that not every choice of a robust function works well [21] and a proper robust function is critical for achieving a robust minimization of the reprojection error when the initial parameters are too noisy and outliers are not explicitly eliminated beforehand. Two commonly used robust statistics functions are the *Cauchy* (or Lorentzian) and *Huber* [203] measures:

- Cauchy or Lorentzian cost function

$$\rho(s) = b^2 \log(1 + s^2/b^2) \tag{7}$$

- Huber cost function

$$\rho(s) = \begin{cases} s^2 & \text{if } |s| < b \\ 2\,b\,|s| - b^2 & \text{otherwise} \end{cases} \tag{8}$$

where s is the residual (i.e. reprojection error) in (2) and b is usually one or a fixed user defined value. However, such standard robust error functions are insufficient when directly used in the BA optimization to combat high percentage of outlier feature correspondence errors as described in [21].

We developed a novel robust function which uses the Cauchy loss or robust error function combined with an *adaptive persistency factor* weight for each feature track. The persistency factor provides a weight based on the number of consecutively matched features compared to the set of all tracks, to reduce the effects of outliers in the optimization error metric:

$$\rho_{ji}(s_{ji}, \gamma_j, \mu_F, \sigma_F) = (\frac{\gamma_j}{\mu_F + \sigma_F})^2 \ \log(1 + (\frac{\mu_F + \sigma_F}{\gamma_j})^2 \ s_{ji}^2) \tag{9}$$

where

$$b = \frac{\gamma_j}{\mu_F + \sigma_F} \tag{10}$$

is the Cauchy scale factor, $s_{ji} = \|\mathbf{x}_{ji} - g(\mathbf{X}_j, \mathbf{R}_i, \mathbf{t}_i, \mathbf{K}_i)\|^2$ denotes the residual of the j-th 3D point in the i-th camera (i.e. feature track), γ_j is the persistency factor related to j-th 3D point, and μ_F and σ_F are the mean and standard deviation of the persistency factor, respectively, for the population of feature tracks. Substituting (9) into (2) we obtain a new robust error function which leads to the global minimization over all feature tracks and views:

$$E_{BA4S} = \min_{\mathbf{R}_i, \mathbf{t}_i, \mathbf{X}_j, \mathbf{K}_i} \sum_{i=1}^{n} \sum_{j=1}^{m} \left\{ (\frac{\gamma_j}{\mu_F + \sigma_F})^2 \cdot \log \left(1 + (\frac{\mu_F + \sigma_F}{\gamma_j})^2 \|\mathbf{x}_{ji} - g(\mathbf{X}_j, \mathbf{R}_i, \mathbf{t}_i, \mathbf{K}_i)\|^2 \right) \right\}. \tag{11}$$

The proposed robust function is inspired by the Cauchy or Lorentzian robust function [104, 203] which has an influence function very similar to the Geman-McClure robust function [149] that decreases rapidly reducing the effect of large outlier values. The residuals, s_{ji} in (9) associated with each feature track are *weighted adaptively*, with longer lived feature tracks being favored (larger γ_j) over residuals with shorter length feature tracks (smaller γ_j). So a larger persistency weight favors longer co-visibility features that are more likely to represent the same real 3D structure point in the scene. The proposed adaptive persistency factor weights using the modified Cauchy robust function in (9) performed the best compared to the standard Cauchy or Huber robust functions *without persistency* [21].

3.2 Georegistration using Global Homography

Vehicle detection and tracking is one of the primary demands in WAMI aerial imaging. To improve detection and tracking in aerial imagery in which videos are captured on a moving platform, the images are registered to maintain the relative movement between the moving platform and the scene fixed. Traditionally, aerial image registration methods are performed through applying 2D homography transformations in the *image space* [130, 99, 120]. Aerial image registration is challenging for urban scenes where there are large 3D structures (tall buildings) causing high amount of occlusion and parallax. In such situations, the presence of parallax can lead to significant error when image space 2D registration approaches are used [58]. A method to register aerial images is developed in this project that utilizes 3D information, particularly camera poses available from the proposed fast and robust camera poses optimization technique, to speed up the process and eliminate the effects of strong parallax and occlusions. Unlike the traditional registration methods, the proposed technique uses special homography transformations which are directly computed (derived as closed-form analytic expression) from available precise 3D camera poses and are not estimated using image-to-image estimation techniques.

3.2.1 Related Work

The majority of approaches for mosaicing and image stabilization use image-based matching and warping either global or piece-wise local transformation for stabilizing the ground plane prior to moving object detection [130, 99, 120, 141, 183, 137, 129, 54, 201, 207, 198, 90, 154]. As mentioned earlier and stressed in [58, 224], aerial image registration is challenging for urban scenes where there are large 3D structure and tall buildings causing high amount of occlusion and parallax. An aerial image registration method was proposed in [130, 105] which uses a multi-layer (coarse to fine) homography estimation approach to deal with parallax and occlusions. Although using a hierarchical homography estimation helped to reduce false registration, their approach still suffer from the presence of strong parallax, as it was not able to seamlessly register all images within a dataset altogether. By observing Table-I in their paper([130]) one can see that each dataset was broken into several segments in the registration process, due to an inability to faithfully handle strong parallax. Molina and Zhu [141] proposed a method to register nadir aerial images in which a pyramid block-based correlation method was used to estimate inter-frame affine parameters. They stated not being able to use available GPS/INS measurements and just relying on the imagery itself: *"the measures made by GPS/INS devices come with errors due to hardware, and [if] used directly will produce panoramas with large apparent errors and discontinuities"*. Their approach requires persistent (multiple) cyclic video data collections to work. Moreover their approach has been only tested on nadir imagery with negligible parallax issues, while in oblique imagery (WAMI) the parallax is significantly stronger. Direct georeferencing of high-resolution unmanned aerial vehicles (UAV) imagery was discussed in [205] while performances of different SfM softwares (Photoscan [3], Pix4D [7] and Bundler [6, 196]) were evaluated. A dataset comprising 143 images of $5{,}184 \times 3{,}456$ pixels was used. The authors concluded that Photoscan has the best performance as it is fast, easy to use and accurate. Photoscan, Pix4D and Bundler each took 4.3 hours, 11 hours and 41 hours, respectively, to run on this dataset. The best achieved result (4.3 hours) in this experiments which is from Photoscan is yet very far away from our result on a similar dataset ("Columbia", 202 aerial images of $6{,}600 \times 4{,}400$ pixels– see our Table 2) which took less than two minutes to run the full pipeline. Pritt from Lockheed Martin [180] proposed a fast orthorectification method for registration of thousands of aerial images (acquired from small UAVs). The author argued that BA is not able to handle hundreds of aerial images and therefore it is not scalable. According to that, a technique was introduced as a replacement to BA for the registration process. The best performance was reported to be *250 images per hour*, or about *14 sec per image*, for a dataset including 4,500 images with sizes of $4{,}000 \times 6{,}000$ pixels. This experiment is comparable with our "Columbia-II*, MO" dataset (see Table 2 in this paper) which includes 5,322 images with size of $6{,}600 \times 4{,}400$ pixels. Compared to our method, which took only 41 min and 45 sec for 5,322 images (0.47 sec per image), Pritt's method is more than 30 times slower than our BA4S. Notice that the results presented in [180] appear to be tested over relatively flat terrain with negligible parallax. Crispell et al. introduced an image registration technique to deal with parallax, assuming to have a dense 3D reconstructed model

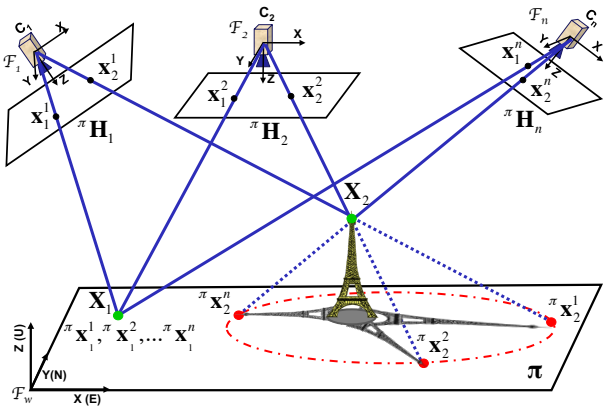


Figure 7: Concept of parallax in aerial imagery. A scene and its dominant ground plane π is observed by n aerial cameras. ${}^{\pi}\mathbf{H}_i$ represents the homography transformation between the image plane of $\mathcal{C}_i$ and the plane π. For an on-the-plane 3D point such as $\mathbf{X}_1$, its homographic transformations from the camera image planes onto π will all merge together and coincide to $\mathbf{X}_1$ (the green point). However, for an off-the-plane point such as $\mathbf{X}_2$, its homographic transformations will be spread out (see the red points ${}^{\pi}\mathbf{x}_2^1$, ${}^{\pi}\mathbf{x}_2^2 \ldots {}^{\pi}\mathbf{x}_2^n$). These red points are spurious and induced by the parallax.

of the scene [58]. In [181] GPS and IMU were used to perform an initial (coarse) orthorectification and georeferencing of each image in an aerial video. Then a RANSAC-based method was used to find optimal affine transformations in 2D image space. A method for registering and mosaicking multi-camera images was proposed in [99]. In the proposed method, registration is achieved using control points and projective image-to-image transformations (using a variation of RANSAC). Among the corresponding control points found in overlapping images, those that best satisfy the projective constraint are used to register the images. Their system took about 15 minutes to estimate the transformation matrices for registering just six overlapping images, each of size $4{,}008 \times 2{,}760$, which is considerably very slow. A technique for optimal feature matching was recently proposed in [135], called locally linear transforming (LLT), which tries to tackle outliers using the consistency of local features within a neighborhood. A local geometrical constraint was developed that can preserve local structures among neighboring feature points which is also robust to a large number of outliers. It has a relatively high complexity and also uses exhaustive iterative methods; two drawbacks in the feature matching stages of a SfM pipeline which we are avoiding in our proposed approach. A similar algorithm, called Restricted Spatial Order Constraints (RSOC), was proposed in [131] to deal with outliers for registering aerial images. Both local structure and global information were considered in RSOC. It assumes that neighbor spatial order is preserved after rigid and affine transformation and based on that an affine invariant descriptor was defined. However such assumption for oblique aerial imagery of urban scenes is not held, due to existence of high parallax. Lee et al. introduced a method for image registration of airborne LiDAR, hyper-spectral and photographic imagery of wooded landscapes [120]. A nonparametric (NP) image registration technique was proposed to align images obtained from multi-sensor imaging. The developed method was mainly for the registration of three types of airborne remote sensing data sampled over wooded landscapes.

3.2.2 Georegistration Technique

Using the previously described approach to refine camera parameters one can globally register the images in geo-reference system. Fig. 7 illustrates a world coordinate system $\mathscr{F}_w$ and a ground plane π spanning through its X and Y axes. Without loss of generality, we consider $\mathscr{F}_w$ to be on π. The scene is observed by n aerial cameras $\mathcal{C}_1$, $\mathcal{C}_2$ $\ldots$ $\mathcal{C}_n$. To make the notations succinct, we will omit the camera indices from now on unless otherwise stated. Let us consider a projective camera $\mathcal{C}$, with coordinate system $\mathscr{F}_c$, observing the scene. Using (4), the image coordinate (expressed in homogeneous system) of a general 3D point $\mathbf{X} = [x \;\; y \;\; z]^{\intercal}$ from $\mathscr{F}_w$ projected on the image plane is obtained:

$$\tilde{\mathbf{x}} = \mathbf{K}\,\big[\mathbf{R}\,\mathbf{X} + \mathbf{t}\big] \tag{12}$$

where $\mathbf{K}$ is the camera calibration matrix (intrinsic parameters), $\mathbf{R}$ and $\mathbf{t}$ are respectively the rotation matrix and translation vector from $\mathscr{F}_w$ to $\mathscr{F}_c$. The Z coordinate of a 3D point $\mathbf{X}$ lying on π is zero

yielding:

$$\tilde{\mathbf{x}} = \mathbf{K}\left(\begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3\end{bmatrix}[x \;\; y \;\; 0]^\intercal + \mathbf{t}\right) \tag{13}$$

$\mathbf{r}_1$, $\mathbf{r}_2$ and $\mathbf{r}_3$ being the first, second and third columns of $\mathbf{R}$, respectively. After simplification we have

$$\tilde{\mathbf{x}} = \mathbf{K}\begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t}\end{bmatrix}{}^{\pi}\tilde{\mathbf{x}} \tag{14}$$

where ${}^{\pi}\tilde{\mathbf{x}} = [x \;\; y \;\; 1]^\intercal$ represent the 2D homogeneous coordinates of 3D point $\mathbf{X}$ on π. Indeed, one can consider the term $\mathbf{K}\begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t}\end{bmatrix}$ analogue to a 3×3 homography transformation matrix which maps any 2D point from π onto the camera image plane as:

$$\tilde{\mathbf{x}} = {}^{c}\mathbf{H}_{\pi}\,{}^{\pi}\tilde{\mathbf{x}}. \tag{15}$$

Likewise, a 2D homogeneous image point $\tilde{\mathbf{x}}$ can project back on π as:

$${}^{\pi}\tilde{\mathbf{x}} \;=\; {}^{c}\mathbf{H}_{\pi}^{-1}\,\tilde{\mathbf{x}} \;=\; {}^{\pi}\mathbf{H}_{c}\,\tilde{\mathbf{x}} \tag{16}$$

where ${}^{\pi}\mathbf{H}_c$ is equal to:

$${}^{\pi}\mathbf{H}_c = \begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t}\end{bmatrix}^{-1}\mathbf{K}^{-1} \tag{17}$$

$$= \begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t}\end{bmatrix}^{-1}\begin{bmatrix} f & 0 & u \\ 0 & f & v \\ 0 & 0 & 1 \end{bmatrix}^{-1} \tag{18}$$

$$= \frac{1}{fD}\begin{bmatrix} r_{22}t_3 - r_{32}t_2 & -r_{12}t_3 + r_{32}t_1 & r_{12}\,t_2 - r_{22}t_1 \\ r_{21}t_3 - r_{31}t_2 & r_{11}t_3 - r_{31}t_1 & r_{11}\,t_2 - r_{21}t_1 \\ r_{21}r_{32} - r_{22}r_{31} & r_{11}r_{32} - r_{12}r_{31} & r_{11}r_{22} - r_{12}r_{21}\end{bmatrix} \begin{bmatrix}1 & 0 & -u \\ 0 & 1 & -v \\ 0 & 0 & f\end{bmatrix} \tag{19}$$

where

$$D = r_{11}r_{22}t_3 - r_{11}r_{32}t_2 - r_{12}r_{21}t_3 + r_{12}r_{31}t_2 + r_{21}r_{32}t_1 - r_{22}r_{31}t_1 \tag{20}$$

$$= (r_{21}r_{32} - r_{22}r_{31})t_1 - (r_{11}r_{32} - r_{12}r_{31})t_2 + (r_{11}r_{12} - r_{12}r_{21})t_3. \tag{21}$$

The columns of rotation matrix $\mathbf{R} = \begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3\end{bmatrix}$ are orthogonal, i.e. $\mathbf{r}_1 \times \mathbf{r}_2 = \mathbf{r}_3$, or

$$\begin{aligned} r_{21}r_{32} - r_{22}r_{31} &= r_{13} \\ -r_{11}r_{32} + r_{12}r_{31} &= r_{23} \\ r_{11}r_{12} - r_{12}r_{21} &= r_{33} \end{aligned} \tag{22}$$

Using (22), after simplification we obtain:

$${}^{\pi}\mathbf{H}_c = \frac{1}{\lambda}\begin{bmatrix} r_{22}t_3 - r_{32}t_2 & -r_{12}t_3 + r_{32}t_1 & -[r_{22}t_3 - r_{32}t_2, & -r_{12}t_3 + r_{32}t_1, & -r_{12}t_2 + r_{22}t_1]\,\mathbf{v} \\ -r_{21}t_3 + r_{31}t_2 & r_{11}t_3 - r_{31}t_1 & -[-r_{21}t_3 + r_{31}t_2, & r_{11}t_3 - r_{31}t_1, & r_{11}t_2 - r_{21}t_1]\,\mathbf{v} \\ r_{13} & r_{23} & & -\mathbf{r}_3^\intercal\mathbf{v} & \end{bmatrix} \tag{23}$$

where $\mathbf{v} = \begin{bmatrix}u & v & f\end{bmatrix}^\intercal$ and λ is a scalar defined as

$$\lambda = f\mathbf{r}_3^\intercal\mathbf{t}. \tag{24}$$

λ in (23) can be omitted since a homography matrix is defined up-to-scale. As a result, we obtain the following analytical expression for the georegistration homography matrix:

$${}^{\pi}\mathbf{H}_c = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33}\end{bmatrix} \tag{25}$$

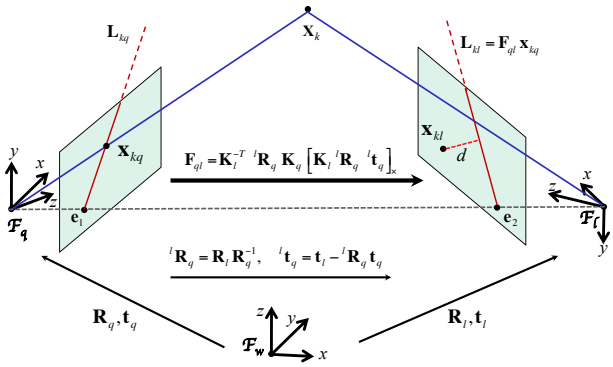


Figure 8: Computation of a fundamental matrix $\mathbf{F}_{ql}$ corresponding to the images of q-th and l-th cameras using their intrinsic and extrinsic parameters. $\mathbf{x}_{kq}$ and $\mathbf{x}_{kl}$ are two corresponding image points (related to a 3D point $\mathbf{X}_k$). Due to noise in camera parameters or feature point correspondences, the fundamental constraint ($\mathbf{x}_{kl}^T\,\mathbf{F}_{ql}\,\mathbf{x}_{kq} = 0$) may not be held, yielding to an error d.

where its elements h_{ij} are as in (23) and we note the following simplification for h_{13} and h_{23}:

$$\begin{aligned} h_{13} &= -\left[h_{11},\ h_{12},\quad r_{12}t_2 - r_{22}t_1\right]\mathbf{v} \\ h_{23} &= -\left[h_{21},\ h_{22},\ -r_{11}t_2 + r_{21}t_1\right]\mathbf{v}. \end{aligned} \tag{26}$$

A homography relationship is always induced by a 3D plane in the scene (in our case the georegistration ground plane π). In the noise free case (for the camera pose parameters), the back projection errors for points on the ground plane will be zero – that is the back projections of the (2D) image points from each camera, of a 3D point lying on the scene plane π, will map to this same 3D point. However, back projection of 2D image points associated with a 3D scene point that is off-the-ground-plane (not lying on π) will be scattered lying on a contour that depends on the height of the object and follows the trajectory of the camera (i.e. clockwise example shown by the red rays in Fig. 7). This is referred to as motion induced parallax. For example, an on-the-plane 3D point such as $\mathbf{X}_1$ has homographies or back projections from each of the cameras ${}^{\pi}\mathbf{x}_1^1,\ {}^{\pi}\mathbf{x}_1^2 \ldots\ {}^{\pi}\mathbf{x}_1^n$, that coincide with the true location $\mathbf{X}_1$. However, for off-the-plane points such as $\mathbf{X}_2$ its homographies or back projections will be spread out as shown by the points ${}^{\pi}\mathbf{x}_2^1,\ {}^{\pi}\mathbf{x}_2^2 \ldots\ {}^{\pi}\mathbf{x}_2^n$. The motion induced parallax of such tall structures will produce apparent motion in the georegistered image sequence once the images are stabilized using the direct homographies. The same approach can be extended to work with a camera array made up of multiple focal planes on the same aerial platform or on multiple platforms in a sensor network.

3.3 Accuracy Evaluation Using Fundamental Matrix-based Euclidean Epipolar Error (EEE)

In aerial WAMI reconstructions, it is not always practical to provide a quantitative evaluation of the results due to a lack of available 3D ground-truth that is both expensive and difficult to collect [82]. Generally, the reprojection error is commonly used for evaluating SfM results. However, in our pipeline the standard L_2 reprojection error might not be a merit metric to compare it against other SfM methods, since outliers are not explicitly eliminated in our approach. Here all spurious scene points as well as valid 3D points across all cameras will contribute to the reprojection error, while our primary objective is to assess the accuracy of the recovered camera pose. Therefore for comparison, we use a pixel-based error measure, to evaluate the SfM results, which uses the classical epipolar constraint to evaluate the quality of the refined camera parameters on the image plane using manually tracked feature points. We call this method as Fundamental matrix-based Euclidean Epipolar Error, or *EEE*. For evaluation purposes, we generate image-based ground truth by manually tracking N_g feature tracks, in each sequential or WAMI dataset preferably over all cameras or views; typically N_g = 11 to 50. Given reference camera, q, then for each possible camera pair (q, l), their corresponding *fundamental transformation matrix* [95] can be analytically computed (without using image point correspondence-based estimation) using camera parameters (i.e. metadata). Consider a pair of cameras, $\mathcal{C}_q$ and $\mathcal{C}_l$, with associated projection matrices $\mathbf{P}_q$ and $\mathbf{P}_l$, and centers $\mathbf{C}_q$ and $\mathbf{C}_l$, respectively. Assume a scene 3D point $\mathbf{X}_k$ is projected on the image planes of $\mathcal{C}_q$ and $\mathcal{C}_l$ respectively as $\mathbf{x}_{kq}$ and $\mathbf{x}_{kl}$. Given the 2D image point $\mathbf{x}_{kq}$, we can determine the set of 3D points in the scene that

map to this point. Such a set will constitute a ray in the scene passing through $\mathbf{C}_q$. Two points on this ray are already known – one is the camera center $\mathbf{C}_q$ (notice that $\mathbf{P}_q\mathbf{C}_q = 0$) and the second point is $\mathbf{P}_q^+\mathbf{x}_{kq}$, where $\mathbf{P}_q^+$ denotes the pseudo-inverse of $\mathbf{P}_q$ (i.e. $\mathbf{P}_q\,\mathbf{P}_q^+ = \mathbf{I}$). Point $\mathbf{P}_q^+\,\mathbf{x}_{kq}$ lies on the ray since it projects to $\mathbf{x}_{kq}$ as $\mathbf{P}_q(\mathbf{P}_q^+\,\mathbf{x}_{kq}) = \mathbf{x}_{kq}$. These two points, $\mathbf{C}_q$ and $\mathbf{P}_q^+\mathbf{x}_{kq}$, are imaged by the second camera $\mathcal{C}_l$ at $\mathbf{P}_l\mathbf{C}_q$ and $\mathbf{P}_l\mathbf{P}_q^+\mathbf{x}_{kq}$, respectively. These two imaged points in the second view (camera $\mathcal{C}_l$) generate the *epipolar line*,

$$\mathbf{L}_{kl} = (\mathbf{P}_l\mathbf{C}_q) \times (\mathbf{P}_l\mathbf{P}_q^+\mathbf{x_{kq}}), \tag{27}$$

The projection of the camera center $\mathbf{C}_q$ in the second view, or point $\mathbf{P}_l\mathbf{C}_q$, is the *epipole* of $\mathcal{C}_q$ (see $\mathbf{e}_2$ in Fig. 8). The epipolar line in (27) can be expressed as

$$\begin{aligned}\mathbf{L}_{kl} &= (\mathbf{P}_l\mathbf{C}_q) \times (\mathbf{P}_l\mathbf{P}_q^+)\mathbf{x_{kq}} \\ &= [\mathbf{e}_2]_\times(\mathbf{P}_l\,\mathbf{P}_q^+)\,\mathbf{x_{kq}} \\ &= \mathbf{F}_{ql}\,\mathbf{x_{kq}}\end{aligned} \tag{28}$$

where $\mathbf{F}_{ql}$ is the *fundamental matrix* which maps any 2D image points from camera $\mathcal{C}_q$ to the image plane of $\mathcal{C}_l$, and defined by

$$\mathbf{F}_{ql} = [\mathbf{e}_2]_\times(\mathbf{P}_l\,\mathbf{P}_q^+) \tag{29}$$

Notice that if $\mathbf{a} = [a_1,\ a_2,\ a_3]^T$ is a 3-vector, then $[\mathbf{a}]_\times$ defines the corresponding 3×3 *skew-symmetric* matrix as follow [93]:

$$[\mathbf{a}]_\times = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \tag{30}$$

The fundamental matrix in (28) can be developed to be expressed by the camera parameters (metadata). Without loss of generality, assume the coordinate system of camera $\mathcal{C}_q$, $\mathscr{F}_q$, as the world origin. Also assume that ${}^l\mathbf{R}_q$ and ${}^l\mathbf{t}_q$ are respectively the rotation matrix and translation vector from $\mathscr{F}_q$ to $\mathscr{F}_l$,

$$\mathbf{P}_1 = \mathbf{K}_q[\mathbf{I}_{3\times1}|\mathbf{0}_{3\times1}], \text{ and } \mathbf{P}_2 = \mathbf{K}_l[{}^l\mathbf{R}_q|{}^l\mathbf{t}_q]. \tag{31}$$

Substituting $\mathbf{P}_q$ and $\mathbf{P}_l$ in (29) respectively with $\mathbf{P}_1$ and $\mathbf{P}_2$ defined in (31) gives [95],

$$\begin{aligned}\mathbf{F}_{ql} &= [\mathbf{P}_l\mathbf{C}_q]_\times(\mathbf{P}_l\mathbf{P}_q^+) \\ &= [\mathbf{K}_l\,{}^l\mathbf{t}_q]_\times\,\mathbf{K}_l\,{}^l\mathbf{R}_q\,\mathbf{K}_q^{-1} \\ &= \mathbf{K}_l^{-\intercal}[{}^l\mathbf{t}_q]_\times\,{}^l\mathbf{R}_q\,\mathbf{K}_q^{-1} \\ &= \mathbf{K}_l^{-\intercal}\,{}^l\mathbf{R}_q[{}^l\mathbf{R}_q^\intercal\,{}^l\mathbf{t}_q]_\times\,\mathbf{K}_q^{-1} \\ &= \mathbf{K}_l^{-\intercal}\,{}^l\mathbf{R}_q\,\mathbf{K}_q^\intercal[\mathbf{K}_q\,{}^l\mathbf{R}_q^\intercal\,{}^l\mathbf{t}_q]_\times\end{aligned} \tag{32}$$

The rotation matrix and translation vectors between the two camera coordinate systems, ${}^l\mathbf{R}_q$ and ${}^l\mathbf{t}_q$, are obtained using their rotation matrices and translation vectors from the metadata expressed in $\mathscr{F}_w$:

$$\begin{aligned}{}^l\mathbf{R}_q &= \mathbf{R}_l\,\mathbf{R}_q^\intercal \\ {}^l\mathbf{t}_q &= \mathbf{t}_l\ -\ \mathbf{R}_l\,\mathbf{R}_q^\intercal\,\mathbf{t}_q\end{aligned} \tag{33}$$

From (33) and (32) we derive the following analytical expression for the fundamental matrix between the image planes of two cameras:

$$\mathbf{F}_{ql} = \mathbf{K}_l^{-\intercal}\,\mathbf{R}_l\,\mathbf{R}_q^\intercal\ \mathbf{K}_q^\intercal\ [\mathbf{K}_q\,\mathbf{R}_l\,\mathbf{R}_q^\intercal\,(\mathbf{t}_l\ -\ \mathbf{R}_l\,\mathbf{R}_q^\intercal\,\mathbf{t}_q)]_\times. \tag{34}$$

For the k-th 3D groundtruth point, $\mathbf{g}_{kl}$, with $k = 1 \ldots N_g$, each projected into camera, l, its corresponding epipolar line induced by $\mathbf{g}_{kq}$ from $\mathcal{C}_q$ is computed and plotted in the reference frame l. The sum of the perpendicular Euclidean distances (see d in Fig. 8) between each epipolar line and its associated ground-truth point, averaged over all points, is used as the error measure between any pair of cameras:

$$\epsilon_{ql} = \frac{1}{N_g}\sum_{k=1}^{N_g} d(\mathbf{g}_{kl}\,,\,\mathbf{F}_{ql}\,\mathbf{g}_{kq}) \equiv EEE \tag{35}$$

This error (referred to as *EEE*) is computed over all possible pairs of cameras in the sequence, $\{(q,l)|\, q,l = 1\,..\,n\}$. Ideally ϵ_{ql} should be zero due to the fundamental geometric constraint. However, the triple product $\mathbf{g}_{kl}^{\intercal}\,\mathbf{F}_{ql}\,\mathbf{g}_{kq} \neq 0$, in real scenarios due to errors in either point correspondences or camera parameters. The errors ϵ_{ql} can be treated as a matrix and visualized using colored picture elements (pels). In addition to computing, ϵ_{ql}, between cameras, q and l, the mean μ_ϵ and standard deviation σ_ϵ of the error over all cameras is:

$$\mu_\epsilon = \frac{1}{n^2}\sum_{q=1}^{n}\sum_{l=1}^{n}\epsilon_{ql}\ ,\ \sigma_\epsilon = \sqrt{\frac{1}{n^2}\sum_{q=1}^{n}\sum_{l=1}^{n}(\epsilon_{ql}-\mu_\epsilon)^2} \tag{36}$$

3.3.1 Evaluating Feature Matches Without Image-based Ground-Truth

Normally for real data, specially large aerial images, it is not feasible to have image-based ground truth for evaluating the features and the matches among them. Sometimes it is very useful to apply different sort of feature extraction algorithms and different kind of matching schemes and evaluate their performance relative to each other for specific scenarios. Limited ground-truthing can be done and we have been doing this routinely to evaluate the quality of the camera metadata corrections. However, evaluating the corrected camera parameters, with hundreds to thousands of cameras, without any ground-truth has not been practical so far (to the best of our knowledge). Alternative methods for evaluating the quality of the metadata or comparing the results from different algorithms are needed due to the lack of sufficient ground-truth for a well distributed set of features and their matches or correspondences in large datasets like WAMI. Considering the importance of the topic, we propose to implement and test algorithms that have the potential to alleviate such barriers by automatically evaluating the quality of features/matches with limited or no manual ground-truth and assist the process of identifying accurate point correspondences. The approach is based on using bundle adjusted camera parameters. The optimized parameters will be used as ground-truth (for camera parameters but not features). For each track of corresponding features, the middle frame is considered as the reference. All points of interest (features) from each image in the track is projected onto the reference image using epipolar lines. The Euclidean distance between each epipolar line and its corresponding point in the reference frame is computed as an error value. Ideally such a value should be zero but this is not the case in real scenarios due to various noise processes and modeling errors. The mean and covariance of these error values have high precision and will be considered as a metric to evaluate the quality of features and precision of the matching algorithm.

3.3.2 GUI for BA, Metadata Validation and Ground Truth Creation

The Epipolar Transformation (EpiX) visualization tool is used to monitor the quality of the raw and corrected metadata as this is crucial for successful estimation of dense 3D point cloud reconstructions. Existing tools like VisualSFM have been developed but are customized for different 3D workflows and do not include efficient ground-truthing tools like EpiX with its optimized projection modules. EpiX uses the Qt GUI software development toolkit, which is a cross-platform application framework for enabling the EpiX software to be deployed on Microsoft Windows, Apple OS X and Linux platforms. EpiX also uses the openCV libraries for supporting matrix operations and low level vision modules such as feature operators. EpiX application consists of three main components that are planned to support:

- File loading: a) loading metadata files, b) Loading and displaying image thumbnails
- Structure from motion (SfM): a) Feature extraction & tracking, b) Loop closure, c) Triangulation & Bundle Adjustment preprocessing, d) Bundle Adjustment optimization, e) Alignment with metadata
- Supporting Tools: a) Visual assessment using epipolar lines, b) Plotting errors in camera position, c) Calculating the EEE matrix, d) Applying PMVS (Patch-based Multi-view Stereo) for assessment or initialization, e) Plotting trajectories of points, f) Ground-truth creation, g) Ground-truth validation

In the ground-truth creation module the user is provided with a convenient tool to quickly produce a long tracked sequence of stationary object feature points (such as the corner of a building or roof structures). These ground-truth points can be used to evaluate the improvement in metadata quality as a result

of applying the BA module and checking various projective errors of the manually selected ground-truth points. The user loads images and metadata using EpiX in the first step. The user then manually selects and marks a feature point of interest in two frames well separated in time, that is with a sufficiently large enough baseline, after which the projection algorithm in EpiX predicts and displays the expected position of the feature point within all of the other frames using the epipolar geometry constraint as shown in Figure 9. The ground truth validation module is used to visually inspect the ground truth data points and make any small corrections for creating a reproducible collection of feature points. Users are able to visualize the displacement errors of the marked feature points in the data collection.

3.4 Voting-based Reconstruction

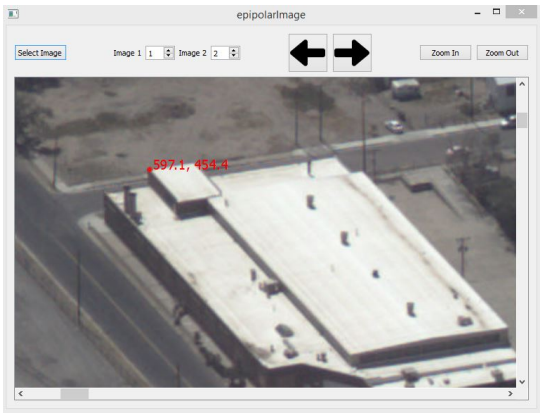
3.4.1 De Novo Voting

Given an accurate camera array sensor model and platform pose and position with respect an earth centered frame of reference we have implemented a voting based algorithm for reconstruction. The votes are accumulated along rays cast from the camera center of projection to the scene for selected interest points identified on the basis of local feature structures. The voxel space geometry is shown in Figure 10. The blue circle shows the flight trajectory, camera coordinate system and retinal plane is shown in red, and the voxel space contains one building in the center of the scene. Rays going from the camera center of projection to the visible corners of buildings faces are shown in black, and the intersection of all of these rays with the ground-plane for one complete circular orbit is shown as green circles. The scale of the building and flight trajectory is exaggerated to facilitate visualization.

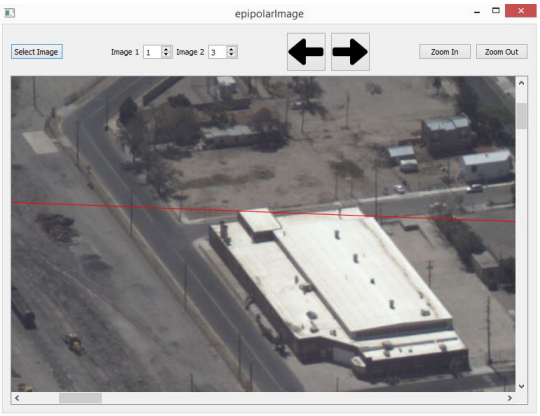
Voting is the first critical part of our 3D reconstruction algorithm and it is distinct from other multi-view approaches that are matching centric. Our voting approach involves covariant feature detection in each frame but no explicit feature matching (for triangulating points). Images are acquired from an aerial platform flying in a circular flightpath at an altitude of approximately 1500 meters (Figure 10). The radius of the circular trajectory is about 3000 meters. For the reconstruction, we use a volumetric approach. A voxel box is computed around the targeted area and we select for each view, distinctive feature points in the image and trace the corresponding ray into the scene. We use ray casting from the camera center of projection, through the voxel box, and vote for voxels along the ray; so a voxel receives as many votes as rays that traverse it. Given a set of images and an accurate estimate of the camera exterior orientations as well as intrinsic camera parameters the voting approach is summarized in Algorithm 1. There are a number of improvements to the basic voting process that are required to improve the quality of the 3D point cloud reconstruction – reducing outlier clusters or false structures that may be floating or attached to other objects, gaps or holes in buildings such as rooftops and walls, vegetation and terrain. Some of these improvements that will investigated are described in the following sections.

Algorithm 1 Voting Algorithm

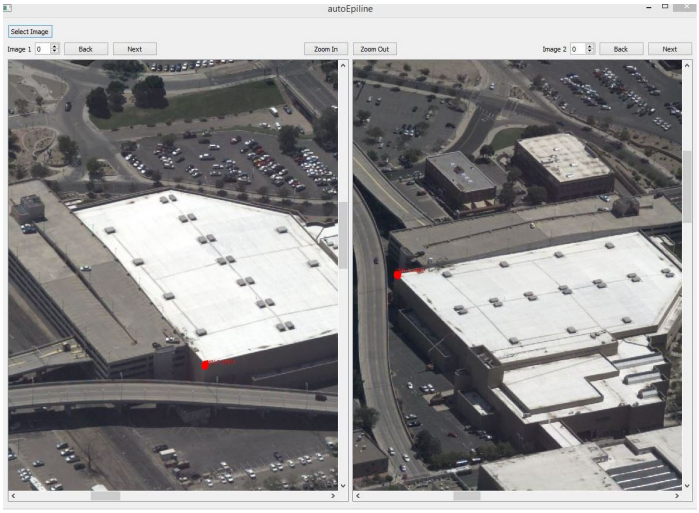
$VoxelBox = InitializeVoxelBox$
for $ImageIndex = 1 : NImage$ **do**
  $FeaturePointList = GetFeaturePointList(ImageIndex)$
  $CameraParameters = GetCameraParameters(ImageIndex)$
  for $FeatureIndex = 1 : NFeatures$ **do**
    $Ray = GetRay(FeaturePointList(FeatureIndex), CameraParameters)$
    $[EntryPoint, ExitPoint] = GetRayAndVoxelBoxIntersections(Ray, VoxelBox)$
    $VoxelList = Bresenham3D(EntryPoint, ExitPoint)$
    for $VoxelIndex = 1 : NVoxel$ **do**
      $VoxelList(VoxelIndex) + +$
    end for
  end for
end for



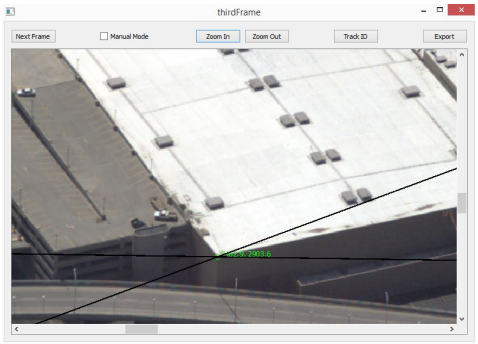
(a) Drawing epipolar line mode - frame t



(b) Drawing epipolar line mode - frame t+1



(c) Ground truth creation mode - Frame t & t+50



(d) Ground truth creation mode - Frame t+1

(e) Frame t ——————- Frame t+1 ——————- Frame t+ 50

Figure 9: Ground truth creation based on marked building feature points in two different views 50 frames apart in time, the intermediate estimated feature point determined semi-automatically, and close up views of the building feature showing the accuracy of user localization (red points) and algorithm estimated location (green points).

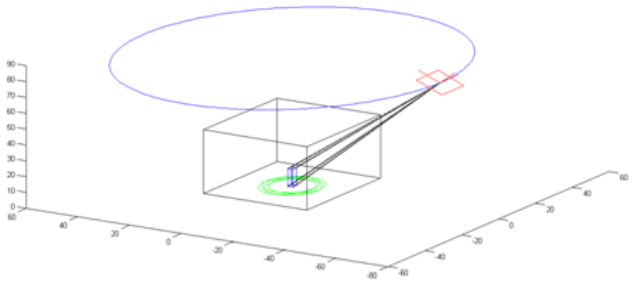


Figure 10: Representation of the flight trajectory (blue circle), camera coordinate system and virtual plane (in red), The voxels box containing one building is in the center of the scene. Rays going from the camera center of projection to the visible corners of buildings faces are represented in black, and the intersection of all of these rays for one complete circular flight with the ground is displayed as green circles. The scale of the building and flight trajectory are not realistic on purpose, in order to facilitate the visualization.

3.4.2 Iterative Voting With Prior Models

The first phase of bundle adjustment optimizes the metadata precision and at the same time also produces a sparse point cloud by triangulating SIFT or other feature point matches, that could be used as prior information for voting. However, this initial structure is not only sparse but also noisy and incorporating noisy observations during the voting process can degrade the reconstruction results. One possible approach would be to compute for each frame a prior depth map that would be used to reduce the length of the rays during voting as shown in Figure 11. This would filter some degree of the noise and obtain a significant speed-up as the Bresenham ray casting is the most expensive computation and its execution time depends directly on the ray length. Furthermore, the process can be repeated in loops: for each new iteration, the denser and more accurate prior depth map is used as a prior constraint, allowing votes to accumulate on more and more features, with shorter and shorter rays. In such an iterative process the voxel resolution could be increased accordingly provided that there is sufficient memory resources to maintain the full volume. The use of geospatial priors and multimodal geoinformatics enables persistent geoint fusion in a consistent manner.

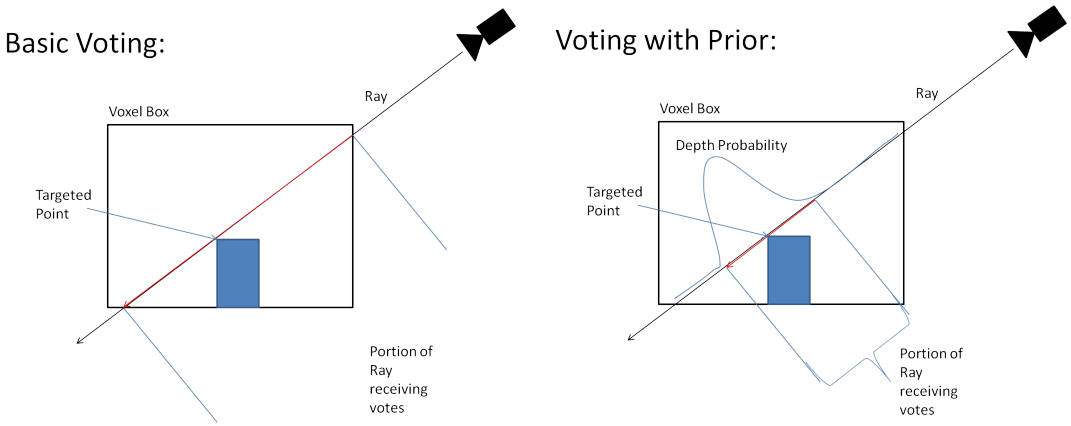
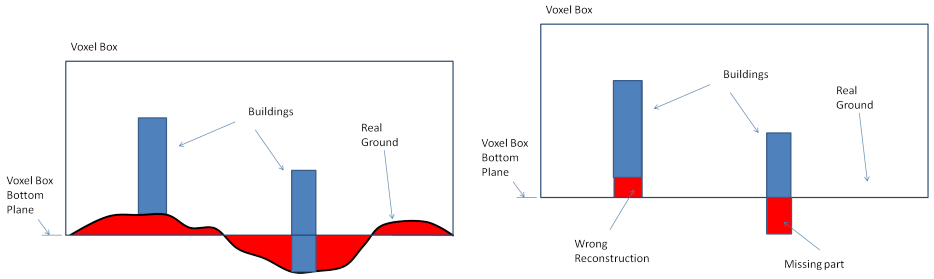


Figure 11: Comparison between the *basic voting* and *voting with priors* approaches. Shortening the portion of the rays that is rasterized will speed up the voting process, and focus the votes within a smaller zone that should reduce the noise and increase the foreground surface signal.

3.4.3 Incorporating Prior Terrain Model Information

Recovering an accurate terrain model is a very challenging task in 3D reconstruction as the terrain usually has less distinctive features than manmade structures. The current algorithm does not incorporate any prior information about the terrain and computes the ground as the lowest horizontal plane of the bounding box.

The bounding box is picked carefully so that its minimum altitude is set to the estimated lowest ground level within the reconstructed geographical region. However, this approximation can be a source of significant error on some datasets since the ground elevation is obviously non-planar within the reconstructed region. Surface elevation information based on geological factors that enable patch similarity to be used to relate semblance in different parts of the scene within the field of view which will enable improved terrain reconstruction. Figure 12 shows a summary of several possible errors. Some terrain information is available through the NASA WorldWind toolkit as well as other sources such as USGS and NGA and will be incorporated as prior information to improve the voting process and the final result. Situation specific priors from previous reconnaissance of the region can also be used. Not only will the reconstruction of the ground elevations be more accurate than current results, but it would also improve several other parts of the pipeline such as voting, vote collapsing or extrusion, computation of prior depth maps and automatic determination of bounding box parameters. The alignment of height fields from different sources needs to be precise to enable fusion of these different sources of 3D information and will likely require matching tie points and ground control points.



(a) Real world structures (b) Expected inaccurate reconstruction

Figure 12: Depending on whether the bounding box ground plane is too high or too low, the reconstruction algorithm might miss some parts of the structure, or the vote collapsing might result in incorrectly reconstructed supporting structures.

3.4.4 Coarse-to-fine Vote Spaces

It may be beneficial to use a multiscale coarse-to-fine voting approach to support higher resolution voxel densities. A coarse set of voxels, with fewer cameras, can be processed very quickly to yield rough estimates of the 3D structure. The size of the voxels and number of cameras can be iteratively refined in a coarse-to-fine approach to increase the resolution of the voxel space and improve the details in the point cloud. Algorithms and spatially adaptive data structures, like octrees, for supporting a coarse point cloud as an initialization to refine to the next finer voxel density level need to be developed and made computationally efficient as well as scalable using out-of-core methods.

3.4.5 Multiview Incremental Vote Accumulation and Removal

Voting all along a ray, from its entry to its exit across the voxel grid, leads to an accumulation of votes throughout the vote volume and contributes to a significant amount of noise in the final vote volume impacting downstream processes surface detection and extraction. This *vote noise* increases for each new view. One potential approach to reduce this vote noise would be to remove some of the votes that we can readily classify as noise early in the voting process – that is subtracting votes from one view (the earliest in a group), while at the same time we are adding votes from another new camera (the most recent view in a group). This leads to an accumulate-remove (or add-subtract) type of voting algorithm to ensure that each voxel receives only one vote (or at most a few) per frame. We propose investigating a modified voting algorithm that enforces an implicit temporal consistency of votes and filters the noise from older frames or views while at the same time the algorithm adds votes from new frames or views. The alternating phases of vote accumulation and vote removal are described in Algorithm 4. Furthermore, this method also enforces that each voxel can receive at most one vote per frame, regardless of the voxel resolution

and feature density as described in Algorithms 2 and 3. Current testing using adjacent views (one second sampling) will be extended to use well separated views such as four second sampling.

Algorithm 2 Voting_Unique (CurrentImageIndex)

```
for All Voxel do
    Voxel.CurrentVote = false
end for
FeaturePointList = GetFeaturePointList(CurrentImageIndex)
CameraParameters = GetCameraParameters(CurrentImageIndex)
for each FeaturePoint in FeaturePointList do
    Ray = GetRay(FeaturePoint, CameraParameters)
    [EntryPoint, ExitPoint] = GetRayAndVoxelBoxIntersections(Ray, VoxelBox)
    RayVoxelList = Bresenham3D(EntryPoint, ExitPoint)
    for each Voxel in RayVoxelList do
        Voxel.CurrentVote = true
    end for
end for
for All Voxel do
    if Voxel.CurrentVote == true then
        Voxel.VoteCount++
    end if
end for
```

Algorithm 3 Voting_Removal (CurrentImageIndex)

```
for All Voxel do
    Voxel.CurrentVote = false
end for
FeaturePointList = GetFeaturePointList(CurrentImageIndex)
CameraParameters = GetCameraParameters(CurrentImageIndex)
for each FeaturePoint in FeaturePointList do
    Ray = GetRay(FeaturePoint, CameraParameters)
    [EntryPoint, ExitPoint] = GetRayAndVoxelBoxIntersections(Ray, VoxelBox)
    RayVoxelList = Bresenham3D(EntryPoint, ExitPoint)
    for each Voxel in RayVoxelList do
        if Voxel.VoteCount < RepeatabilityThreshold then
            Voxel.CurrentVote = true
        end if
    end for
end for
for All Voxel do
    if Voxel.CurrentVote == true then
        Voxel.VoteCount--
    end if
end for
```

3.4.6 Automatic Point Cloud Extraction

Our automatic point cloud extraction requires as a first step computation of the vote histogram so that we can transform the vote to a normalized value for each voxel. The first part is histogram computation which is straightforward; we simply compute the frequency (pdf) and cumulative (cdf) histograms of values for both votes and the associated gradient magnitude estimates which provide information about scene structure. The size of the bins is set to one. An example of the histograms for the Albuquerque dataset are shown in Figure 13, 14, and 15. These include the vote and vote gradient histograms on linear and log scales, the vote and vote gradient histograms after vote extrusion to improve building structures, vote gradient histogram after vote extrusion and Gaussian smoothing to filter out noisy votes and a 2D histogram of votes versus vote gradients. These histograms demonstrate the difficulty in trying to select a single manual or automatic vote threshold to separate foreground from background since there is no bimodal structure between the two classes as seen in Figure 19.

Algorithm 4 Alternating Vote Accumulation and Vote Removal

1: **for** $FrameIndex$=1 to $RepeatabilityThreshold$ **do**
2: Voting_Unique($FrameIndex$)
3: **end for**
4: **for** $FrameIndex$=$RepeatabilityThreshold + 1$ to N_Frames **do**
5: Voting_Removal($FrameIndex - RepeatabilityThreshold$)
6: Voting_Unique($FrameIndex$)
7: **end for**

Fusing Information for Automatic Vote Thresholding: The number of votes and the range of the gradient magnitudes depend on a number of parameters including number of frames used for voting and the grid resolution. In case of automatic thresholding it becomes necessary to normalize the information contained in the voxel box. Therefore, we use the histograms to define a low threshold below which the voxel score is set to zero and a high threshold above which the score is set to one and use a sigmoid or logistic function (see Figure 17) to assign a normalized score between 0 and 1 to any value between the low and the high thresholds. The low threshold is chosen as the mean value, and the high threshold as the value such as only 2 percent of the voxels are above it. Two independant scores are computed, one for the number of votes and one for the gradient magnitude and for each voxel the two scores are then multiplied together to obtain a final score normalized between 0 and 1. Usually we retain only those voxels in the final point cloud with a value above 0.5 to 0.8.

3.5 Deriving a Depth Map Probability Distribution from the Vote Volume

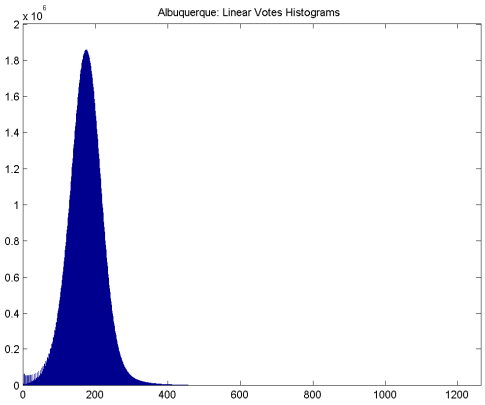
Instead of applying a global threshold, the vote volume can be converted into a per voxel probability of occupancy likelihood map, from which we can then derive a depth distribution along each ray. The order in which the voxels are encountered is important and the first voxel with very high occupancy probability should correspond to the maximum of likelihood for the depth value or location of the surface, even if there exist peaks at voxels with higher vote value further along the ray. We can formalize this constraint as a voxel is visible from a position if and only if all the voxels in between are not occupied. Therefore, given the probability of occupancy for each voxel in between, we can compute the probability of visibility as the product of all preceding voxel probabilities of non-occupancy as shown in Equation 37. Then, the current voxel corresponds to the feature observed in the image if and only if that voxel is visible, and occupied. We consider in that case that the depth of the ray is the distance from the camera center to the current voxel. We can simplify without loss of generality, using $Depth = i$ where i is the voxel index, varying from 1 which is the closest to the camera to N which is the furthest away as shown in Equation 38. Before all of this, the probability of occupancy is computed by normalizing the vote volume, using a sigmoid as shown in Figure 17 and the two boundaries corresponding to a low threshold, taken as the average number of votes, below which the probabiility equals 0 and a high threshold that corresponds to the top 2 percent of the number of votes, above which the probability is set to 1.

$$P_{Visibility}(i) = \prod_{j<i}(1 - P_{Occupancy}(j)) \tag{37}$$

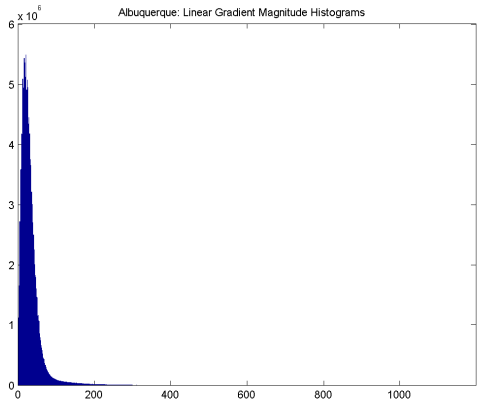
$$P(Depth = i) = P_{Visibility}(i) * P_{Occupancy}(i) \tag{38}$$

3.6 Combining Depth with Appearance Consistency

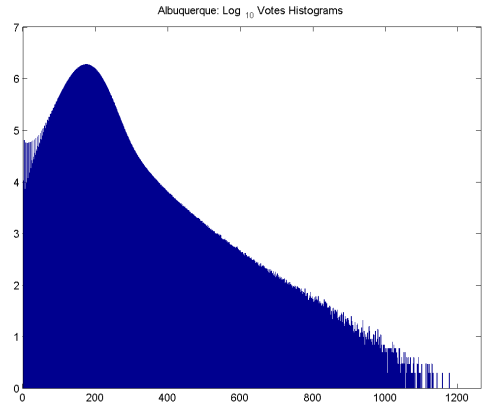
It is possible to make the model more robust by adding appearance information. Each voxel can be projected onto several images, and a matching score can then be computed (using either cross correlation or a descriptor such as ORB or Daisy). If the appearance is not consistent between different views, it should decrease the probability that this voxel is occupied. However, we do not want the consistency information to dominate the previously computed depth probability. So the appearance matching score is normalized in the range $\alpha \in [0, 1]$. A value of zero corresponds to a total inconsistency that should raise the probability of non-occupancy or transparency to one. A high value for consistency on the other hand



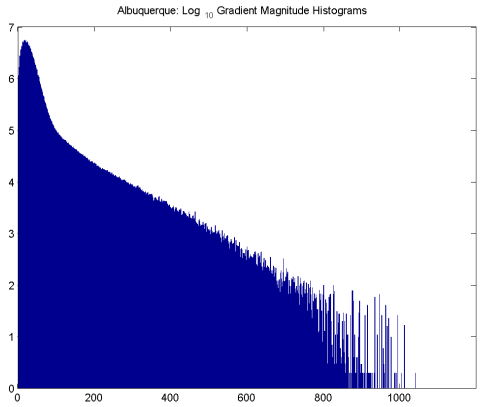
(a) Histogram of votes (vertical axis linear scale)



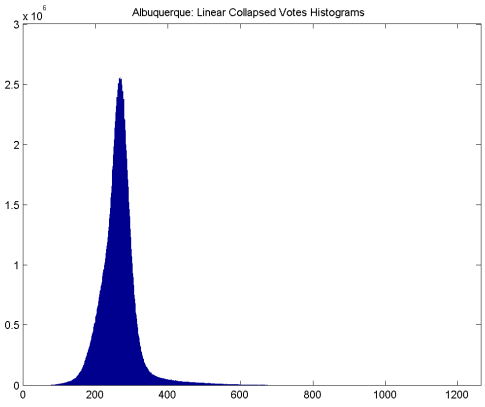
(b) Histogram of vote gradient magnitude (linear scale)



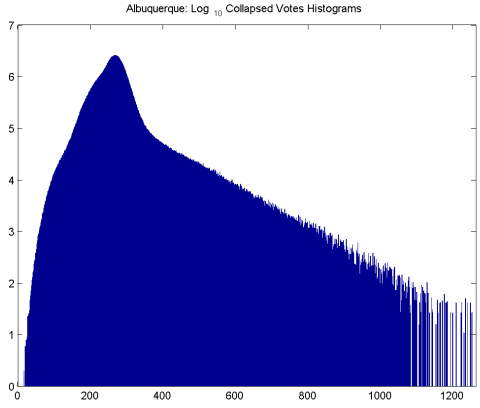
(c) Histogram of votes (vertical axis log scale)



(d) Histogram of vote gradient magnitude (log scale)



(e) Histogram after vote extrusion/collapsing (linear scale)



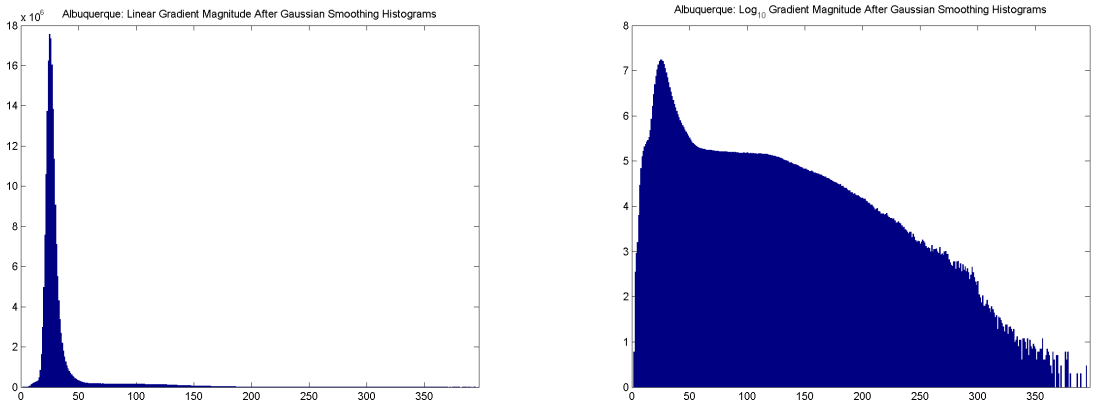
(f) Histogram after vote extrusion/collapsing (log scale)

Figure 13: Example of vote space and vote gradient magnitude histograms for the Albuquerque WAMI dataset.

should decrease the probability of non-occupancy or transparency; but the latter should still depend on the voting information. We therefore propose to use α as an exponent using the following equations:

$$P_{Visibility}(i) = \prod_{j<i}(1 - P_{Occupancy}(j))^{\alpha_j} \tag{39}$$

$$P(Depth = i) = P_{Visibility}(i) * [1 - (1 - P_{Occupancy}(i))^{\alpha_i}] \tag{40}$$



(a) Vote gradient mag after Gaussian smoothing (linear scale) (b) Vote gradient mag after Gaussian smoothing (log scale)

Figure 14: Gaussian smoothing the vote space then computing gradient magnitude histograms for the Albuquerque WAMI dataset filters out a lot of noisy votes.

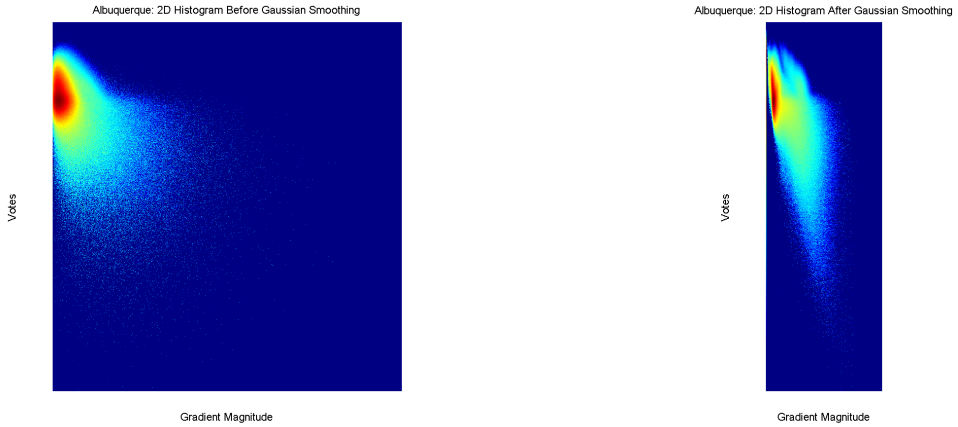


Figure 15: 2-D histograms showing the joint relationship between the vote space and the vote gradient magnitude for the Albuquerque WAMI dataset before and after Gaussian smoothing filters out noisy votes.

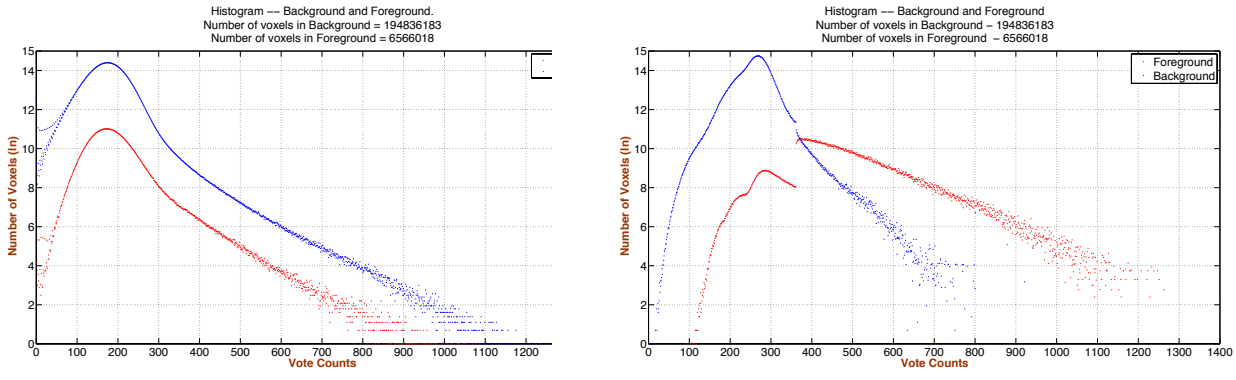


Figure 16: Example of the background vote class shown in blue and foreground surface reconstruction class shown in red before and after vote extrusion operator has been applied. The total overlap of the two classes indicates the difficulty in identifying an appropriate threshold to separate the two classes.

Equation 40 can be simplified to:

$$P(Depth = i) = P_{Visibility}(i) - P_{Visibility}(i+1) \tag{41}$$

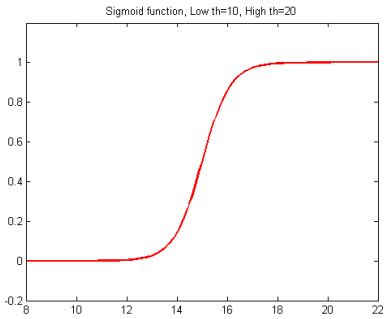


Figure 17: Example of a shifted sigmoid or logistic function that is used to transform vote values into a surface probability map. The low threshold is 10 so that any vote values below 10 is assigned the probability 0, any vote value above 20 is assigned a probability of 1 with a continuous transition between the two thresholds.

3.7 Visualization of Point Clouds and Meshes

3.7.1 Point Cloud Coloring

Once the regularized point cloud geometry has been estimated with improved surface details and holes in surfaces removed, we need to color voxels in the point cloud for rendering visually accurate surface appearance estimates that can be used for visualization and other applications like improving tracking. Several algorithms were investigated since some of the initial coloring algorithms were extremely slow taking on the order of several days. Consistent colors depend on an accurate estimate of surfaces, associated surface normals and material properties. Algorithm 5 selects the voxel color from the view that minimizes the angle between the ray and the local surface normal. Algorithm 6 strives for view consistency and uses as far as possible voxel colors from the same view. A fast coloring algorithm using a z-buffer type occlusion check is being designed to further improve the estimation of surface colors (Algorithm 7).

Algorithm 5 was the very first coloring method that was implemented supporting different parameters. All the images first need to be loaded into memory which is why we could only use it with the Brown University test datasets. The Transparent Sky images were too large to fit into memory. The input must be a point cloud with the calculated normal for each point. During Summer 2013 we were using Meshlab to estimate the normals. The views that minimize the dot product between the ray and the normal are chosen for coloring. If $NSelectedView = 1$, then only one view is picked per point, which is what we tried the first time. This causes some obvious discontinuities in the final results, where a change of view occurs. If $NSelectedView > 1$, then the color assigned to each point is somehow summarizing the colors read from several images. Hence the Statistic function below on line 14 can be the mean, the median or the mode for example. The experiments we did used the mean, and if we were seeing fewer discontinuities, the rendering had a blurry effect. Importantly. this method does not account for occlusion.

Algorithm 5 Coloring Method 1: Select the view minimizing the angle between the ray and surface

1: **for** $PointIndex = 1 : NPoints$ **do**
2: **for** $ViewIndex = 1 : NView$ **do**
3: Ray=$PointCoordinate(PointIndex) - CameraPosition(ViewIndex)$
4: N=$Normal(PointIndex)$
5: $DotProductArray(ViewIndex)$=$Dot\_Product(N, Ray)$
6: **end for**
7: $[SortedArray\ SortedViewIndex]$=$Sort(DotProductArray, increase)$
8: $BestViewList$=$SortedViewIndex(1 : NSelectedView)$
9: **for** $ViewIndex = 1 : NSelectedView$ **do**
10: $SelectedViewIndex$=$BestViewList(ViewIndex)$
11: $[Px\ Py]$=$BackProject(PointIndex, SelectedViewIndex)$
12: $ColorArray(ViewIndex)$=$Image(SelectedViewIndex, Px, Py)$
13: **end for**
14: $Color(PointIndex)$=$Statistic(ColorArray)$
15: **end for**

Algorithm 6 is the method we have been using since the first results on Albuquerque (Fall 2013) and

that we are continuing to use currently. The idea is to color as many point as possible using image colors from the same view. Images are loaded one by one, and all points are projected on it. If the occlusion test is negative and that the projection is within the image frame, the pixel color is assigned to the point, which is marked as colored and will not be subsequently updated again. This method gives an impressive rendering if the point cloud is seen from positions close to the first image's position. However, the point cloud color quality significantly decreases as we get further from the initial position especially with low sun-angles. An additional drawback of his method is that it has relatively slow performance since the check for occlusions requires the use of discrete voxel-based ray-casting.

Algorithm 6 Coloring Method 2: Use as much as possible the same image for neighboring pixels

```
ImageSubset = ExctractImageSubset()
for ImageIndex = 1 : NImagesInSubset do
    for PointIndex = 1 : NPoints do
        if IsColored(PointIndex) == false then
            VoxelList = Bresenham3D(CurrentPoint, CameraCenter, VoxelBox)
            if AllVoxelsAreEmpty(VoxelList) == true then
                ImageProjection = GetImageProjection(ImageIndex, PointIndex)
                Color(PointIndex) = ImageProjection.Color
                IsColored(PointIndex) = true
            end if
        end if
    end for
end for
```

Algorithm 7 should produce a result very close to the results given by Algorithm 6, but is a lot faster as it does not need to call the (Bresenham) ray-casting function to enforce the occlusion constraint. The difficulty is to correctly fix the depth map resolution. Points in the point cloud do not have any radius and their dimension is therefore regulated by the depth map parameters. If this resolution is too high, the depth map will be sparse, and the occlusion check inaccurate. If the depth map resolution is too low, then there will be more discarded points for occlusion than there should have been. This method can also include a cost minimization component. Two types of experiments will be done, one where the function cost is the depth, and another where it will be the gradient magnitude of the depth.

3.7.2 Converting Point Clouds to Mesh Representation

Once we have a sparse or dense point cloud, we can construct a triangulated mesh from the point cloud to extract high quality surfaces in the aerial scene that can be used to improve the quality of the reconstruction and segment building structures. Similar to the challenges we have in tracking for large scale aerial imagery, accurate modeling is a difficult task due to severe occlusions, highly reflective surfaces, varying illumination conditions, registration errors and sensor noise. On the other hand, the modeling performance is highly dependent on the accuracy of the generated point clouds and the presence of outliers. There are several volumetric methods which could be used for surface extraction that can scale up to the large grid size of urban scenes. Marching cube is one of the well-known surface meshing algorithms that is based on polygonizing the grid and is highly parallelizable which makes it a good candidate to convert point clouds to meshes in real-time. The space complexity of triangle vertices and faces lists include: (i) 3 $\times$ sizeof(float) for xyz, (ii) 3 $\times$ sizeof(int) for RGB, (iii) 3 $\times$ sizeof(float) for normals, and (iv) 3 $\times$ sizeof(int) for triangle faces. Storing all of this information requires an efficient data structure for the large scale volumes of our reconstructed urban scenes. The data structure must also have the property of efficient dynamic updates in order to insert new (unique) triangle vertices. Using a hash-table does not appear to be scalable in terms of fast access due to the large number of vertices. We have tried three different approaches using hash tables including sorted and unsorted maps, which are C++ templated data structures, and also a separate hash table to map the unique grid edges and its corresponding cut-point which technically represents a triangle vertex. In the proposed new approach we use a self-adjusting splay tree with the assumption that the most recently accessed elements will most probably be accessed again making the splay tree an appropriate data structure to consider. Therefore in the process of polygonizing

Algorithm 7 Coloring Method 3: Fast Coloring using a Z-Buffer type occlusion check

$ImageSubset = ExctractImageSubset()$
$DepthMap$=$assign < float > (DepthMapWidth, DepthMapHeight)$
$IndexMap$=$assign < int > (DepthMapWidth, DepthMapHeight)$
$Cost$=$assign < float > (NPoints)$
$Cost.fill(INF)$
for $ImageIndex = 1 : NImagesInSubset$ **do**
  $DepthMap.fill(INF)$
  $IndexMap.fill(-1)$
  $CurrentImage = LoadImage(ImageIndex)$
  for $PointIndex = 1 : NPoints$ **do**
    [px py $Depth$]=$BackProject(PointIndex, ImageIndex)$
    [x y]=$Image2DepthMap(px, py)$
    if $Depth < DepthMap(x, y)$ **then**
      $DepthMap(x, y) = Depth$
      $IndexMap(x, y) = PointIndex$
    end if
  end for
  for $x = 1 : DepthMapWidth$ **do**
    for $y = 1 : DepthMapHeight$ **do**
      if $IndexMap(x, y)! = -1$ **then**
        $CurrentPointIndex$=$IndexMap(x, y)$
        $CurrentCost$=$CostFunction(x, y)$
        [px py]=$DepthMap2Image(x, y)$
        if $CurrentCost < Cost(CurrentPointIndex)$ **then**
          $Cost(CurrentPointIndex) = CurrentCost$
          $Color(CurrentPointIndex) = CurrentImage(x, y).Color$
        end if
      end if
    end for
  end for
end for

the grid cells the most recent inserted vertices will be kept in the first levels of the tree and will boost the vertex search performance if it has been already added before. Mesh simplification and thickening, hole closing and texturing are the other issues that need to be addressed after reconstructing the building facades. One solution to fill the holes is to merge adjacent grid cells. In this way we may lose some structure information (like small windows) during merging. This can be avoided if we add some additional contextual feature and structure information as constraints during the merge process. An example mesh reconstructed from a portion of the Los Angeles point cloud is shown in Figure 18.

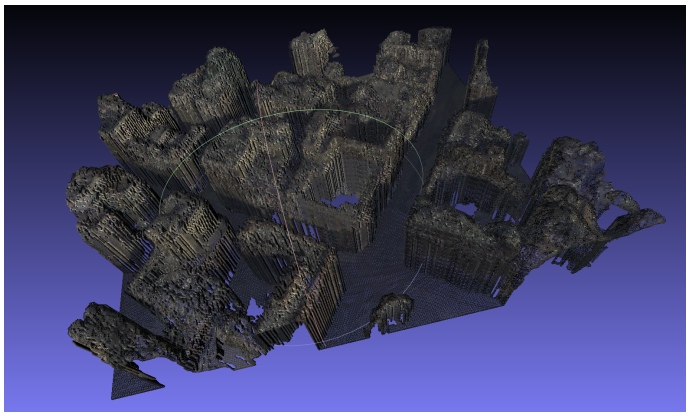


Figure 18: Triangular meshing of the reconstructed 3D point cloud for a small portion of the Lost Angeles dataset.

3.7.3 Out-of-core Spatial Data Structures for Scalability to Large Point Clouds

Extremely large point clouds will usually exceed the primary memory of most computers, especially laptops and mobile devices. In order to process and display such massive point clouds an out-of-core strategy is needed. The proposed approaches serialize the in-memory streaming octree to secondary storage. One method for serializing the streaming octree to disk leverages the hierarchical nature of the file system. Each node in the octree can be represented by a directory in the filesystem. Each child node (sub-octree) is a subdirectory of the parent. Additionally, each directory would contain a single file with the actual point cloud data.

Top-most node of a streaming octree showing child nodes as subdirectories and the data file containing the coarsest sampled view of the entire point cloud. The figure shows one level of a serialized streaming octree. The data file contains a fixed number of points from the point cloud representing a sampled view of the current octant. Each subdirectory represents a nested octree containing possibly more children but also another collection of points that refine the parent. If any of the eight sub-octrees are empty, then no subdirectory will exists; this is the serialized equivalent of a null child from the in-memory octree. Traversing this serialized octree is the same as traversing the in-memory version. Here pointers to nested octrees are just directories. The data file itself contains the point positions, colors, normal, votes, histograms, etc. The data files could be compressed to reduce storage and improve read performance. A single metadata file for the entire octree (not shown) is all that is needed in addition to the file hierarchy. This metadata file should contain the bounding box of the entire point cloud, which point attributes are contained in the data files and their order, and optional compression type.

Unlike the above approach which requires many directories and files, a second method for serializing the streaming octree produces a single file. This approach represents pointers to nodes as offsets in the file. Following metadata describing the point data (attributes, compression, and maximum extents), an offset to the top most node is provided. This offset is absolute indicating how many bytes from the beginning of the file.

Each node contained in the file contains eight file offsets indicating where in the file those child nodes reside on disk. Following the offsets to child nodes is the actual point cloud data (possibly compressed) for the current node. For any empty octants, the file offset will be the special value zero; because of the header and metatdata no nodes can exist at the very beginning of the file. This zero offset parallels a null pointer of the in-memory octree. This approach is nearly identical to the in-memory data structure except instead of memory pointers to sub-octrees, the pointers are represented as file offsets.

An important consideration for both approaches to serializing the data to disk is that none of the octants are directly available for reading; rather finding the data to read requires traversing several steps either through the file system or several seeks in the single file approach. While this might be considered a disadvantage this actually closely mirrors the process used when displaying the point cloud data in Stratus. No single node contains all the data for a given octant. Each parent contains a coarse view and the children nodes refine that data of the parent. To draw all points contained in an octant, all the points from the root to the leaf node is required. This hierarchical refinement is at the core of efficient culling and continuous level-of-detail display of data in Stratus.

3.7.4 Point Cloud Refinement

In order to improve the understanding of 3D structures in the scene extraction of building footprints, the ground plane and identifying the connected components related to buildings will be necessary. We use a plane fitting approach with RANSAC to filter out noise in identifying the largest fitting plane to identify the ground plane and then subsequently identifying the various separate buildings in the scene. Some preliminary results in this direction are shown in Figure 19.

3.8 Moving Vehicle Detection using Semantic Depth Map Fusion

Automatic moving object detection and segmentation is one of the fundamental low-level tasks for many of the urban traffic surveillance systems including traffic monitoring, activity and behavior recognition

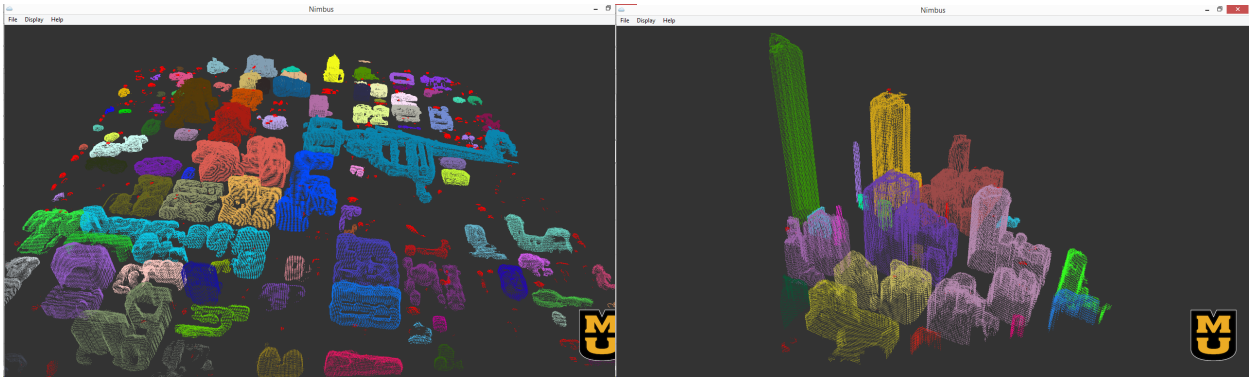


Figure 19: Initial segmentation of building structures using RANSAC-based plane fitting for a portion of Albuquerque (left) and Los Angeles (right).

and tracking applications. We developed an automatic moving vehicle detection system for wide area aerial imagery based on semantic fusion of flux trace and building mask information. Flux trace provides spatio-temporal information of moving edges including false detected moving buildings caused by parallax effects. We deployed building depth information to filter out motion detection due to parallax effect from actual moving vehicles on the ground. The coarse building depth map were guided towards the actual building boundaries using a level-set geodesic active contour framework. An average precision of 90% and recall of 80% have been reported for 200, $2k \times 2k$ cropped region of interest from Albuquerque urban imagery.

3.8.1 Introduction

Automated video-based supervision systems known as video surveillance systems are essential for monitoring of indoor or outdoor scenes and activities. An efficient management of transport networks, traffic-signal monitoring, protection of infrastructures, border control, health care monitoring applications and many others [177, 125]. Automatic moving object detection and segmentation is one of the fundamental low-level tasks for many of urban traffic surveillance system applications including traffic monitoring[128], activity and behavior recognition [193, 145, 76, 51], change detection [200], classification [144, 143, 194, 74, 192] and in particular detection and tracking [213, 162, 219, 36, 177, 111, 164, 179, 190, 101, 55, 150, 165, 176].

Detection and segmentation of moving objects from stationary background is challenging/complicated due to background variance, illumination changes, shadow or varying camera viewpoints [125, 101, 163, 69]. Detection and tracking in airborne urban traffic surveillance is impacted by many more difficulties such as small and low resolution targets, large moving object displacement due to low frame rate, congestion and occlusions, motion blur and parallax effect, camera vibration and exposure and varying viewpoints [185, 200, 70, 55, 128, 71, 169, 166, 160].

A typical moving object detection system follows either an appearance-based or motion-base approaches. Moving vehicle detection algorithms typically focus on motion-based detection methods [48, 190, 72] since appearance-based methods [31, 192] are mostly computationally expensive particularly when applied to large scale aerial imagery. However, many recent methods deploy fusion schemes to combine the strength of each individual detection technique and improve system robustness [125, 109, 72, 213, 210, 119, 77].

In this project, we developed a motion-based detection technique exploiting fusion of flux trace spatio-temporal information and building depth map to enhance the robustness and filter out false responses caused by tall buildings parallax effects. The coarse building masks were refined and guided to the actual building boundaries using a level-set geodesic active contour framework to save the detected moving vehicle next to the buildings. We used the state of the art registration algorithm called *MU BA4S* in order to orthorectified image sequences in a global reference system to maintain the relative movement between the moving camera platform and the fixed scene [20, 22]. Figure 20 illustrates our proposed semantic fusion-based moving vehicle detection system pipeline.

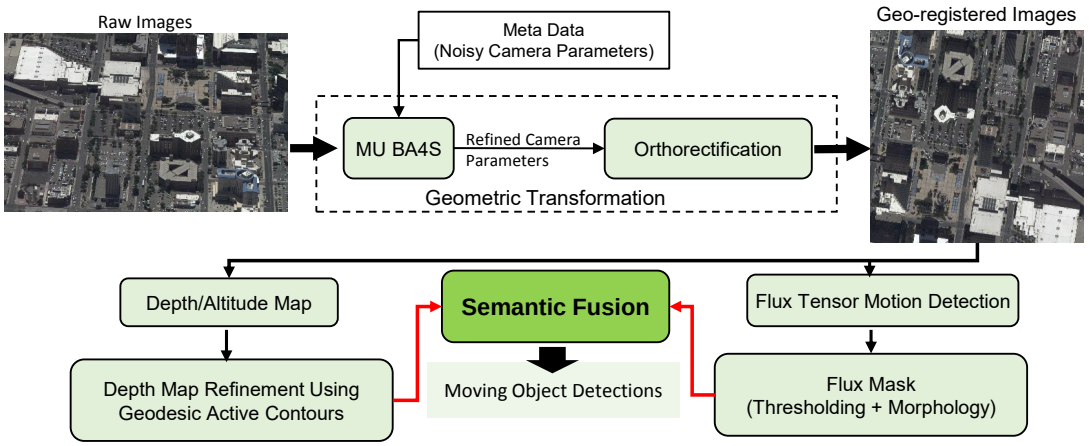


Figure 20: Semantic fusion-based moving vehicle detection system pipeline.

3.8.2 Orthorectification

Videos in aerial imagery are captured on a moving airborne platform. Detection of moving objects, e.g. vehicles, in a scene which is observed by a camera that by itself has large movement and big jitters can be extremely challenging. To address this problem, images from camera planes are orthorectified (registered) in a global reference system to maintain the relative movement between the moving platform and the scene fixed. In this work the registration has been carried out by applying a homography transformation between each image plane and the ground dominant plane of the scene. Such homography transformations are analytically obtained using camera parameters, i.e. their rotation matrices and translation vectors. First the noisy camera parameters (referred to as *metadata*) obtained from platform sensors (i.e. IMU and GPS) are refined by a fast Structure-from-Motion algorithm(BA4S), proposed in [20, 22]. After the refinement process, the homography matrix between the ground plane π and the image plane of camera C is computed as follows:

$$ {}^{c}\mathbf{H}_{\pi} = \mathbf{K} \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t} \end{bmatrix} \tag{42} $$

$\mathbf{r}_1$ and $\mathbf{r}_2$ being the first and second columns of the rotation matrix from the world's to the camera local coordinate system, $\mathbf{t}$ is the corresponding translation vector, and $\mathbf{K}$ is the camera calibration matrix. As a result, a 2D homogeneous image point $\tilde{\mathbf{x}}$ from camera C can project back on π as:

$$ {}^{\pi}\tilde{\mathbf{x}} = {}^{c}\mathbf{H}_{\pi}^{-1}\, \tilde{\mathbf{x}} = {}^{\pi}\mathbf{H}_{c}\, \tilde{\mathbf{x}} \tag{43} $$

where ${}^{\pi}\mathbf{H}_{c}$ is the inverse of the homography ${}^{c}\mathbf{H}_{\pi}$ in 42. Such a homography transformation is valid to transform all points between the image and ground plane, provided that their corresponding 3D point lies on the ground plane. Otherwise applying this homography transformation for points off the ground plane would cause a phenomenon known as *parallax* (see [94] for more details).

In this part of the project, we conducted our experiments on ABQ aerial imagery which were collected by TransparentSky [5] over downtown Albuquerque, NM and ortho-rectified using *MU BA4S* state-of-the-art registration approach. Figure 21 shows samples of raw and geo-registered ultra high resolution images (6400×4400). However, we worked on a cropped $2k \times 2k$ region of interest for which the ground-truth are provided by our collaborator at Kitware (Fig. 21(BR)).

3.8.3 Depth Maps in Orthorectified Projections

The output of our MU BA4S bundle adjustment produces refined camera parameters and a sparse 3D point cloud [20, 22, 19]. Sparse point clouds can be improved to produce dense point clouds using different multi-view stereo algorithms like PMVS[83], VisualSFM/Bundler[218] or probabilistic volume method described in [57]. The dense 3D point clouds are then used to produce depth or altitude maps by projecting the 3D points into each camera view. Each point is projected on a grid on the image plane, and after taking occlusion into account, each pixel in the grid is assigned at most one point in the point cloud.

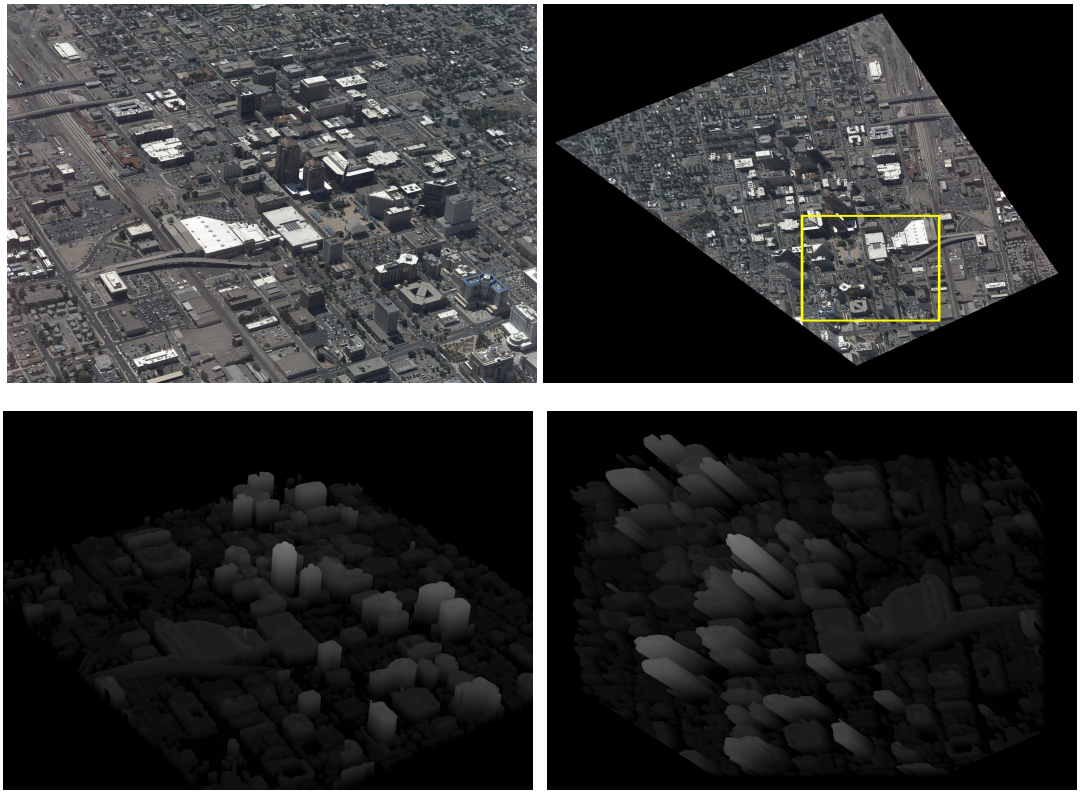


Figure 21: (UL) Illustrates raw ultra high resolution images ($\sim$ 30 MB, each) collected from an airborne platform flying over downtown Albuquerque-NM. (UR) corresponding raw image depth mask (LL) shows the corresponding registered images exploiting *MU BA4S* orthrectification approach, (LR) corresponding orthorectified depth mask

The altitude value of this point is used as the intensity for the corresponding pixel. This way we can produce a mask for each view such as the one shown in Figure 21(LL). Then same homography that was applied to original image can be used for the altitude mask to obtain the estimated altitude of every pixel in the ortho-rectified image, as shown in Figure 21(LR). Frame specific depth maps provide valuable information for distinguishing between motion detections due to motion parallax (motion of tall structures due to viewpoint changes) and moving objects on the ground. The semantic fusion method for combining depth and motion is discussed in more detail in Section 3.8.6.

3.8.4 Motion Detection

The developed framework uses *flux tensor* to extract motion blobs and to refine building mask. Flux tensor is presented as an extension of 3D structure tensor that allows reliable motion segmentation without expensive eigenvalue decomposition [45, 157]. Following section briefly describes flux tensor structure and computations.

3.8.5 Flux Tensors

Flux tensor provides reliable detection of moving structures by computing temporal variations of the optical flow field within the local 3D spatiotemporal volume. Under constant illumination model, optic-flow equation of a spatiotemporal image volume $\mathbf{I}(\mathbf{x})$ centered at location $\mathbf{x} = [x, y, t]$ [100] is

$$\begin{aligned} \frac{d\mathbf{I}(\mathbf{x})}{dt} &= \frac{\partial \mathbf{I}(\mathbf{x})}{\partial x} v_x + \frac{\partial \mathbf{I}(\mathbf{x})}{\partial y} v_y + \frac{\partial \mathbf{I}(\mathbf{x})}{\partial t} v_t \\ &= \nabla \mathbf{I}^T(\mathbf{x})\, \mathbf{v}(\mathbf{x}) \end{aligned} \tag{44}$$

taking the derivative of Eq. 44 with respect to t, we obtain Eq. 45

$$\begin{aligned}\frac{\partial}{\partial t}\left(\frac{d\mathbf{I}(\mathbf{x})}{dt}\right) &= \frac{\partial^2\mathbf{I}(\mathbf{x})}{\partial x\partial t}\,v_x + \frac{\partial^2\mathbf{I}(\mathbf{x})}{\partial y\partial t}\,v_y + \frac{\partial^2\mathbf{I}(\mathbf{x})}{\partial t^2}\,v_t \\ &\quad + \frac{\partial\mathbf{I}(\mathbf{x})}{\partial x}\,a_x + \frac{\partial\mathbf{I}(\mathbf{x})}{\partial y}\,a_y + \frac{\partial\mathbf{I}(\mathbf{x})}{\partial t}\,a_t \end{aligned} \tag{45}$$

which can be written in vector notation as,

$$\frac{\partial}{\partial t}(\nabla\mathbf{I}^T(x)\mathbf{v}(\mathbf{x})) = \frac{\partial\nabla\mathbf{I}^T(\mathbf{x})}{\partial t}\,\mathbf{v}(\mathbf{x}) + \nabla\mathbf{I}^T(\mathbf{x})\,\mathbf{a}(\mathbf{x}) \tag{46}$$

where $\mathbf{v}(\mathbf{x}) = [v_x, v_y, v_t]$ is the optic-flow vector and $\mathbf{a}(\mathbf{x}) = [a_x, a_y, a_t]$ is the acceleration of the image brightness located at $\mathbf{x}$. Usually $\mathbf{v}(\mathbf{x})$ is estimated by minimizing Eq. 46 over a local 3D image patch $\boldsymbol{\Omega}(\mathbf{x}, \mathbf{y})$:

$$\frac{\partial\nabla\mathbf{I}^T(\mathbf{x})}{\partial t}\,\mathbf{v}(\mathbf{x}) + \nabla\mathbf{I}^T(\mathbf{x})\,\mathbf{a}(\mathbf{x}) = 0 \tag{47}$$

Assuming a constant velocity model subject to the normalization constraint $||\mathbf{v}(\mathbf{x})|| = 1$ and consequently zero acceleration, a least-squares error measure $e_{ls}(\mathbf{x})$ (Eq. 48) is used to minimize the Eq. 47

$$\begin{aligned}e_{ls}(\mathbf{x}) = \int_{\boldsymbol{\Omega}(\mathbf{x},\mathbf{y})} &\left(\frac{\partial(\nabla\mathbf{I}^T(\mathbf{y})}{\partial t}\,\mathbf{v}(\mathbf{x})\right)^2 d\mathbf{y} \\ &+\lambda\left(1 - \mathbf{v}(\mathbf{x})^T\mathbf{v}(\mathbf{x})\right)\end{aligned} \tag{48}$$

Differentiation of $e_{ls}(\mathbf{x})$ with respect to v, leads to eigenvalue decomposition problem $\mathbf{J_F}(\mathbf{x})\ \hat{\mathbf{v}}(\mathbf{x}) = \lambda\ \hat{\mathbf{v}}(\mathbf{x})$. The 3D flux tensor $\mathbf{J_F}$ for the spatiotemporal volume centered at (x, y) can be written in expanded matrix format as

$$\mathbf{J_F} = \begin{bmatrix} \int_{\boldsymbol{\Omega}}\left\{\frac{\partial^2\mathbf{I}}{\partial x\partial t}\right\}^2 d\mathbf{y} & \int_{\boldsymbol{\Omega}}\frac{\partial^2\mathbf{I}}{\partial x\partial t}\frac{\partial^2\mathbf{I}}{\partial y\partial t}d\mathbf{y} & \int_{\boldsymbol{\Omega}}\frac{\partial^2\mathbf{I}}{\partial x\partial t}\frac{\partial^2\mathbf{I}}{\partial t^2}d\mathbf{y} \\ \int_{\boldsymbol{\Omega}}\frac{\partial^2\mathbf{I}}{\partial y\partial t}\frac{\partial^2\mathbf{I}}{\partial x\partial t}\,d\mathbf{y} & \int_{\boldsymbol{\Omega}}\left\{\frac{\partial^2\mathbf{I}}{\partial y\partial t}\right\}^2 d\mathbf{y} & \int_{\boldsymbol{\Omega}}\frac{\partial^2\mathbf{I}}{\partial y\partial t}\frac{\partial^2\mathbf{I}}{\partial t^2}d\mathbf{y} \\ \int_{\boldsymbol{\Omega}}\frac{\partial^2\mathbf{I}}{\partial t^2}\frac{\partial^2\mathbf{I}}{\partial x\partial t}d\mathbf{y} & \int_{\boldsymbol{\Omega}}\frac{\partial^2\mathbf{I}}{\partial t^2}\frac{\partial^2\mathbf{I}}{\partial y\partial t}d\mathbf{y} & \int_{\boldsymbol{\Omega}}\left\{\frac{\partial^2\mathbf{I}}{\partial t^2}\right\}^2 d\mathbf{y} \end{bmatrix} \tag{49}$$

The elements of the flux tensor incorporate information about temporal gradient changes which leads to efficient discrimination between stationary and moving image features. Thus the trace of the flux tensor matrix which can be compactly written and computed as, $\mathbf{trace}(\mathbf{J_F}) = \int_{\boldsymbol{\Omega}} ||\frac{\partial}{\partial t}\nabla\mathbf{I}||^2 d\mathbf{y}$ can be directly used to classify moving and non-moving regions without the need for expensive eigenvalue decompositions.

3.8.6 Fusion of Flux Trace and Depth-map Information Using Edge Feature Indicator Functions

As described in Section 3.8.4, each pixel is classified as moving versus stationary by thresholding trace of the corresponding flux tensor matrix ($\mathbf{trace}(\mathbf{J_F})$). However, approximately 70% of the motion detection mask are induced by parallax motion affects of tall structures (Fig. 54). We develop a context-based semantic fusion approach to reject such false detections due to tall structures by using the depth map information in a boundary refinement and filtering processing stage. As it is described in Section 3.8.3, the approximate altitude of every pixel in the ortho-rectified image can be computed from dense 3D reconstruction. In order to produce the building mask, image depth map is thresholded. We identify all the pixels with altitude values greater than a defined threshold as (greater than 20 meters for this sequence) as tall structures which will be removed from flux tensor motion mask (Fig. 55). Figure 22 illustrates the true motion detection produced by flux tensor (in yellow color) and false detection caused by parallax in white color. The high altitude areas which are filtered by building mask shown in blue. Provided Ground-truth bounding-boxes are shown in red to visually evaluate the performance of the detection.

2D depth maps are projected from 3D point clouds obtained by 3D reconstruction of the scene (Section 3.8.3). These point clouds have lower resolution compared to the analyzed images. Low resolution

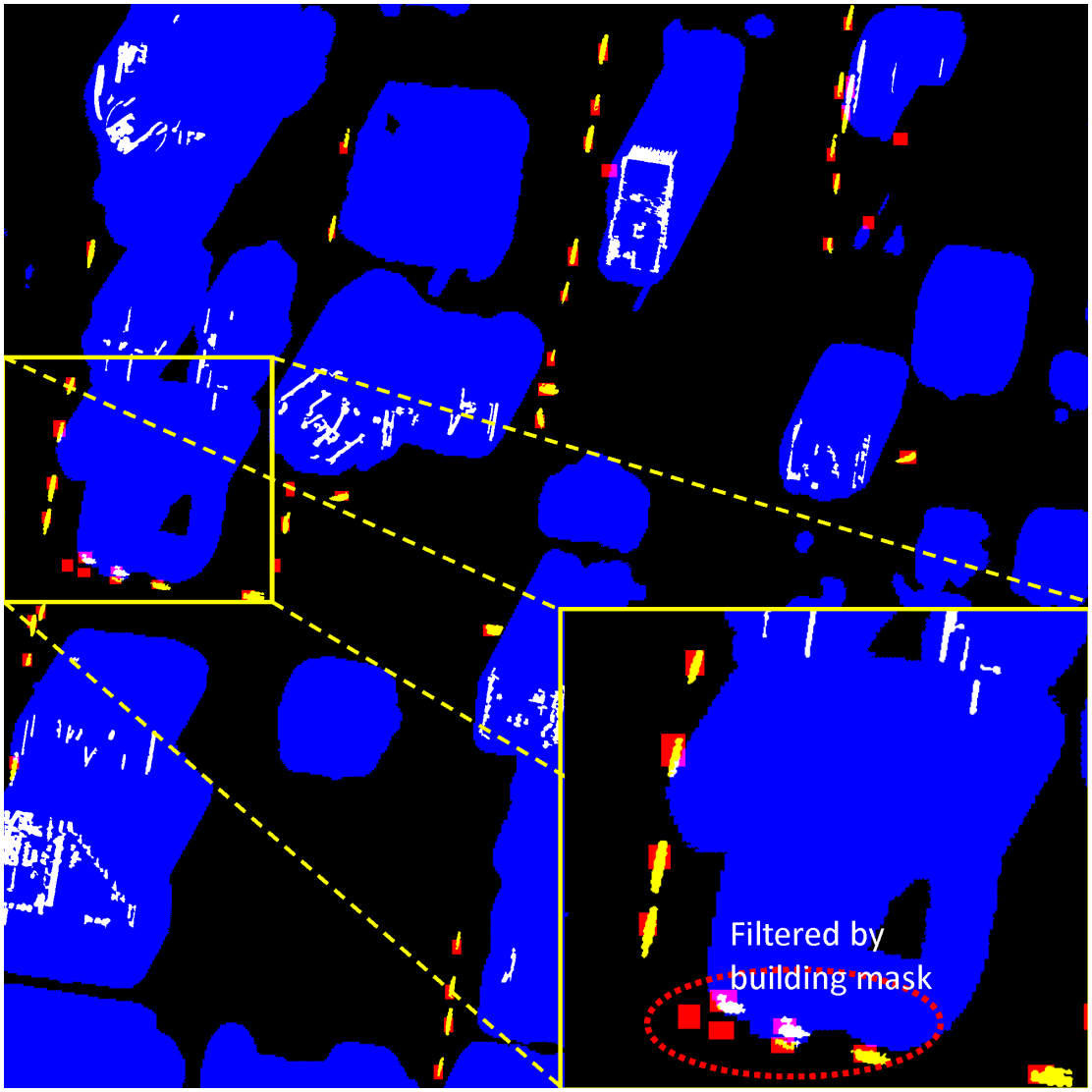


Figure 22: illustration of motion detection results: true motion detection produced by flux tensor (in yellow color) and false detection caused by parallax in white color. The high altitude areas which are filtered by building mask shown in blue. Provided Ground-truth bounding-boxes are shown in red to visually evaluate the performance of the detection.

combined with 2D projection inaccuracies may result in filtering out correctly detected vehicles positioned close to tall structure (Fig. 22).

In order to refine the coarse building map B_{dmap}, we proposed to fuse the high resolution moving edge information from flux tensor trace with building depth mask through a level-set based geodesic active contours framework.

3.8.7 Building Mask Refinement Using Level Set Based Active Contours

The flux trace is used to construct an edge indicator function g_F which will guide and stop the evolution of the geodesic active contour when it arrives at tall structure boundaries,

$$g_F(\mathbf{trace}(\mathbf{J_F})) = \frac{1}{1 + \mathbf{trace}(\mathbf{J_F})} \tag{50}$$

The edge indicator function is a decreasing function of the image gradient that rapidly goes to zero along building edges and holds high values elsewhere.

Active contours evolve a curve $\mathcal{C}$, subject to constraints from a given image. In level set based active contour methods the curve $\mathcal{C}$ is represented implicitly via a Lipschitz function ϕ by $\mathcal{C} = \{(x, y)|\phi(x, y) = 0\}$, and the evolution of the curve is given by the zero-level curve of the function $\phi(t, x, y)$. Evolving $\mathcal{C}$ in

a normal direction with speed F amounts to solving the differential equation [52],

$$\frac{\partial \phi}{\partial t} = |\nabla \phi| F; \quad \phi(0, x, y) = \phi_0(x, y) \tag{51}$$

Unlike parametric approaches such as classical snake, level set based approaches ensure topological flexibility since different topologies of zero level-sets are captured implicitly in the topology of the energy function ϕ. Topological flexibility is crucial for our application since we want to guide the coarse thresholded building mask to the actual building contours and reveal the filtered moving vehicles next to buildings. We used geodesic active contours [49] that are effectively tuned to Flux trace edge information. The level set function ϕ is initialized with the signed distance function of the coarse building mask (B_{dmap}) and evolved using the geodesic active contour speed function,

$$\frac{\partial \phi}{\partial t} = g_F(\mathbf{trace}(\mathbf{J_F}))(c + \mathcal{K}(\phi))|\nabla \phi| + \nabla \phi \cdot \nabla g_F(\mathbf{trace}(\mathbf{J_F})) \tag{52}$$

where $g_F(\mathbf{trace}(\mathbf{J_F}))$ is the fused edge stopping function (Eq. 50), c is a constant, and $\mathcal{K}$ is the curvature term,

$$\mathcal{K} = div\left(\frac{\nabla \phi}{|\nabla \phi|}\right) = \frac{\phi_{xx}\phi_y^2 - 2\phi_x\phi_y\phi_{xy} + \phi_{yy}\phi_x^2}{(\phi_x^2 + \phi_y^2)^{\frac{3}{2}}} \tag{53}$$

The force $(c + \mathcal{K})$ acts as the internal force in the classical energy based snake model. In this work, the constant velocity c pushes the curve inwards to the tall structures. The regularization term $\mathcal{K}$ ensures boundary smoothness. The external image dependent force $g_F(\mathbf{trace}(\mathbf{J_F}))$ is used to stop the curve evolution at building boundaries edges. The term $\nabla g_F \cdot \nabla \phi$ introduced in [49] is used to increase the basin of attraction for evolving the curve to the boundaries of the objects.

Figure 23 shows the improved building contours results in red. The blue line are the initial building contours which are evolved and stopped at building actual boundaries. As it can be seen, the previously filtered detected cars by initial building mask are revealed and counted as true detections.

3.8.8 LoFT: Likelihood of Features Single Object Tracking Framework

Likelihood of Features Tracking (LoFT)[167] is our original recognition-based target tracking system initially designed for vehicle tracking in low frame rate wide area motion imagery, later also used in full motion video analysis. LoFT uses a rich feature set describing intensity, edge, shape and texture information (Figure 24).

Target to search window feature comparison is performed using cross-correlation and sliding window histogram differencing using an efficient integral histogram computation scheme. The process produces a likelihood map for each individual feature. Different features are more sensitive or more robust against different target characteristics and environmental conditions. Fusing different feature likelihood maps enables adaptation of the tracker to scene dynamics and target appearance variabilities. LoFT uses two likelihood map weighting schemes, variance ratio (VR) [56] for histogram-based features and distractor index (DI) [156] for correlation-based features. LoFT appearance adaptation scheme maintains and updates a single template by calculating affine changes in the target to handle orientation and scale changes [167]. LoFT target template update is performed continuously to ensure accurate target localization.

3.9 Dynamic Scene Analysis

On this project, the goal was to perform *dynamic* 3D scene analysis. To that end, beside structural scene analysis, we have focused on moving object tracking for analysis of scene dynamics. Tracking is the process of estimating the locations of objects of interest over time. Despite all the recent advancements in computer vision, visual tracking remains to be a challenging task because of the real-world conditions such as partial or full occlusions, background clutter, shadow and other illumination artifacts.

Many visual tracking approaches have been proposed in the literature to address the unique requirements of different applications. These tracking approaches can be categorized in various ways such

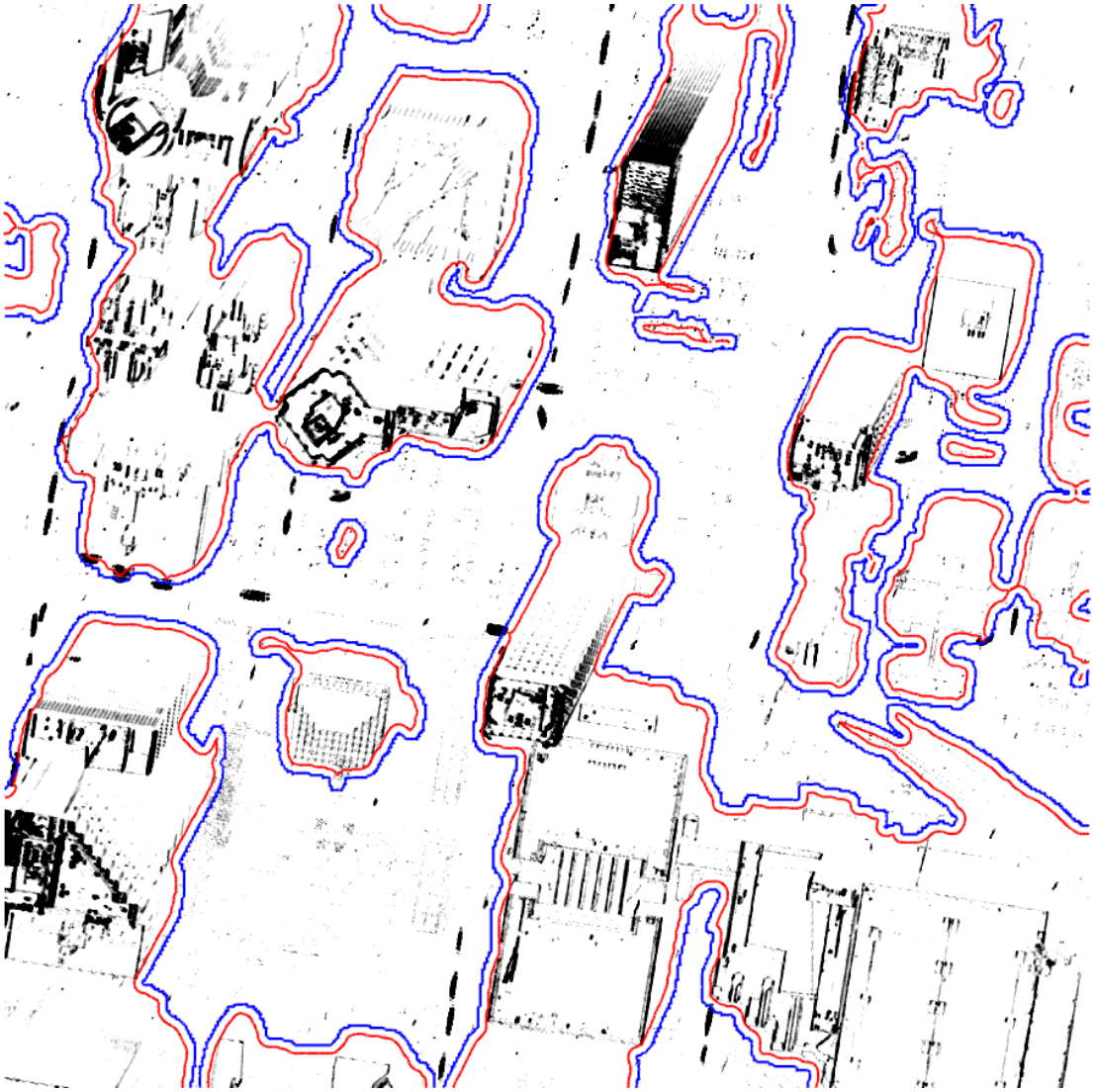


Figure 23: Improved building mask using geodesic active contours: blue lines are the initial building contours which are evolved and stopped at building actual boundaries (red lines). As it can be seen, the previously filtered detected cars by initial building mask are revealed and counted as true detections.

as detection-based [92] [189] versus recognition-based [178][220]; single-object [220][86][158] versus multi-object [11][30][44]; or according to how the objects are localized within the search region, as generative[108][138] versus discriminative methods [126][40].

3.9.1 CS-LoFT: Color and Scale Adaptive LoFT for Single Object Tracking

CS-LoFT extends the original LoFT framework which have shown to obtain good results for WAMI datasets [170][168] [156] with (i) color attributes for improved appearance description; (ii) automated feature scale selection for scale change adaptation; and (iii) an operation mode decision unit to switch between CS-LoFT's three different modes of operation (color, intensity, and motion kinematics) based on appearance reliability. Figure 25 illustrates the operation mode decision with some examples. Color information is incorporated to the LoFT framework using our extension of *Color Name (CN)* scheme introduced in [206]. Max-pooling scale selection is done on likelihood maps obtained for local shape index and normalized curvature index features. The operation mode decision unit activates color use when there is adequate color dissimilarity between the tracked object and its surrounding region. Incorporation of color and scale selection improves the tracker performance particularly for sequences where the tracked objects have distinct color signatures and undergo rapid scale changes.

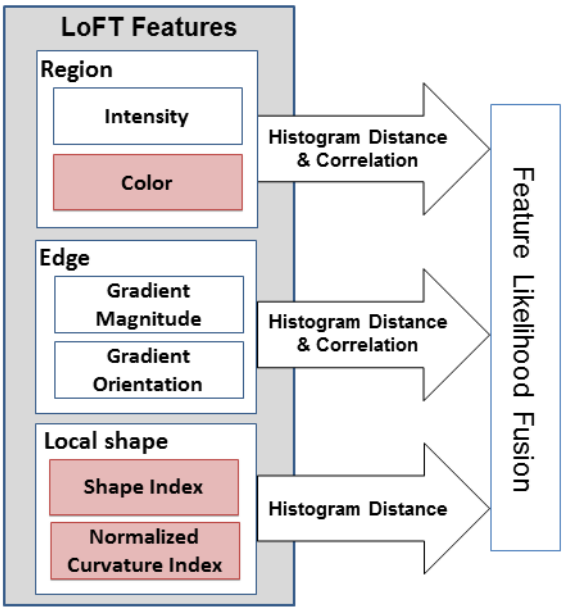


Figure 24: LoFT features.

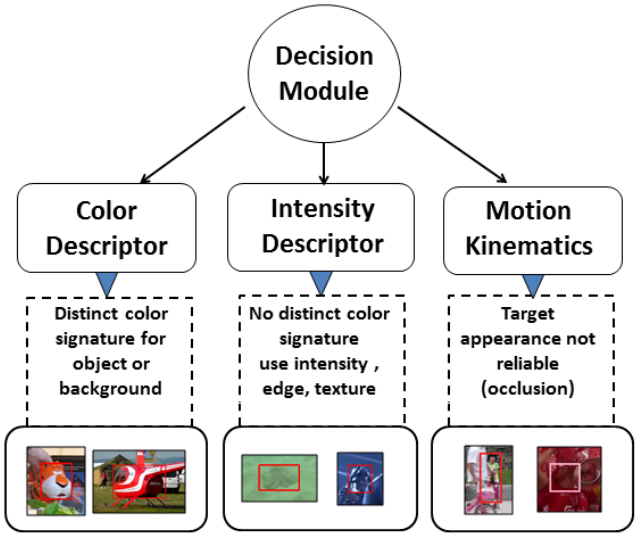


Figure 25: Decision mode operation, with three modes (color, intensity, and kinematics), the tracker switch across these three modes according to the sequence environment.

3.9.1.1 Adaptive Color Use with Color Names (CN) Feature

Most modern trackers ignore color information and rely purely on features derived from grayscale information [91][221][66]. Grayscale images are sometimes sufficient to produce reasonably good tracking results for lower computational cost. However rich chromatic information can be very valuable for object tracking. Various methods have been proposed to incorporate color information to trackers such as simple color space transformations in [172, 153] or color histograms in [225]. More recently, well performing trackers have been further improved with incorporation of color information. In [50] Centinet et al. combined color histograms with Minimum Output Sum of Squared Error (MOSSE) correlation filter [40] as a robust descriptor for object tracking. Danelljan et al. [62] explored the contribution of color in tracking-by-detection frameworks and extended CSK tracker [96] with various color incorporation approaches. Benlefki et al. [32] used color attributes along with motion masks to boost tracker performance. While color introduces rich information for object description, color measurements can vary significantly over an image sequence due to variations in illumination, shadows, and specular reflections. Invariance to these factors has been studied in image classification[96][153], action recognition[96], and tracking [62].

CS-LoFT extends the original LoFT framework with color information using our extension of *Color Names (CN)* feature [206] *(CN-Ex)* . The goal is to boost the performance of the tracker particularly for video sequences where foreground and background have distinct color signatures. Linguistic study described in [33] shows that English language contains eleven basic color terms: *black, blue, brown, grey, green, orange, pink, purple, red, white and yellow*. *Color Names (CN)* feature associates RGB color model with linguistic color labels. We use the mapping provided by [206] to map RGB color models to 10 color names. The process reduces the 256^3 RGB color space to 10^1 color names space. While *CN* includes mapping for black, gray, and white. We add three additional bins for low saturation regions (low saturation & low intensity, low saturation & mediaum intensity, low saturation & high intensity), to prevent their switch from one color name to another with slight illumination changes, shadow etc.

Color information is used in the tracking process only when tracked object and its surroundings have distinct color features. C-LoFT uses an adaptive tracking scheme and uses color, intensity, or motion kinematics depending on availability and reliability of these features. We use Bhattacharyya distance [37] to measure color similarity between target and the surrounding area. Let H_{obj} and H_{bg} be $13 - bin$ color names histograms for tracked object and surrounding background respectively. Discrete probability densities $p(i)$ for the object, and $q(i)$ for the background, are computed by normalizing each histogram by corresponding number of elements in it:

$$p(i) = \frac{H_{obj}(i)}{n_{obj}}; \qquad q(i) = \frac{H_{bg}(i)}{n_{bg}} \tag{54}$$

Bhattacharyya distance between object versus background color distributions is computed as:

$$S = \sum_{i=1}^{13} \sqrt{p(i)q(i)} \tag{55}$$

Inclusion (or exclusion) of color feature is determined for each sequence based on the value of S. Figure 26 shows the process of calculating S for two difference sequence. Low values of S correspond to discriminant color information (dissimilar object versus background color distribution). Figure 27 illustrates the incorporation of color information in C-LoFT.

3.9.1.2 Scale Space Tracking

When tracked objects move along the camera axis, their sizes in the video change. Many trackers such as CSK[96], or MOSSE[40] track the objects of interest at a single scale and ignore possible changes of size. This implies poor performance in sequences with significant scale variations. Incorporation of multi-scale templates as in DSST (Discriminative Scale-Space Tracking) [60] improves tracking performance through scale changes.

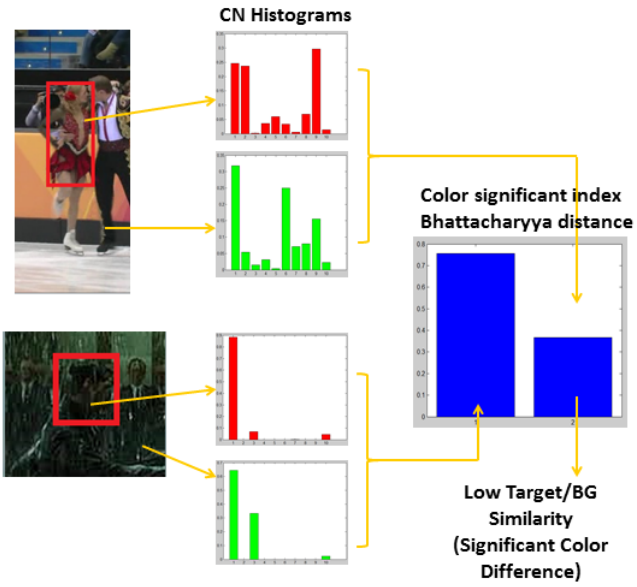


Figure 26: Two examples of different sequences show the difference values of S depends on the similarity between target and surrounded region.

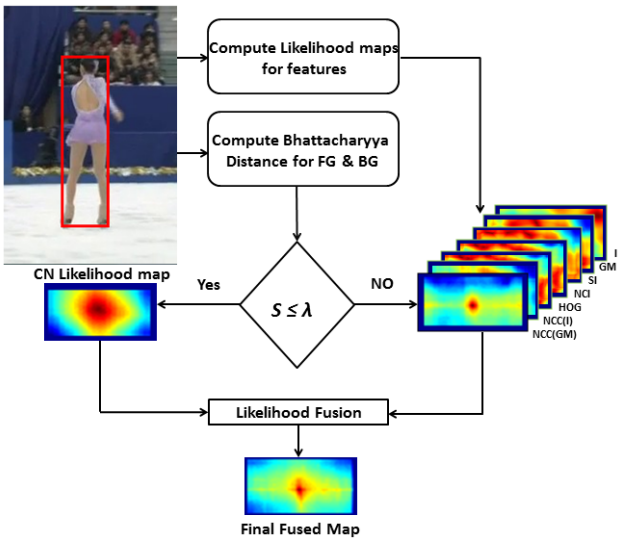


Figure 27: Incorporation of color information in C-LoFT. In which the likelihood map of I represents: Intensity histogram, GM: Gradient magnitude histogram, SI: Shape index histogram, NCI: Normalize curvature index histogram, HoG: Histogram of gradient, $NCC(I)$: Normalized cross correlation for intensity, and $NCC(GM)$: Normalized cross correlation for Gradient

CS-LoFT extends LoFT with multi-scale processing. Scale changes not only alter size of the tracked objects but also affect scale of their texture. LoFT features that are most sensitive to scale changes are local shape features *local shape index* (Eq.58) and *normalized curvature index* (Eq.59) [156] derived from eigenvalues of Hessian matrix.

Hessian matrix H describes the second order structure of local intensity variations around each image point $L(x;y)$,

$$H(x,y;\sigma) = \begin{bmatrix} L_{xx}(x,y;\sigma) & L_{xy}(x,y;\sigma) \\ L_{xy}(x,y;\sigma) & L_{yy}(x,y;\sigma) \end{bmatrix} \tag{56}$$

where $L_{xy} = \frac{\partial^2 L}{\partial x \partial y}$, $L(x,y;\sigma)$ refers to the Gaussian smoothed intensity image

$$L(x,y;\sigma) = g(x,y;\sigma) * L(x,y) \tag{57}$$

g is the Gaussian kernel, and σ is the scale. Local shape index (Eq.58) and normalized curvature index (Eq.59) are derived from the eigenvalues $\lambda_1 \geq \lambda_2$ of Hessian matrix H.

$$\phi(x,y) = \tan^{-1} \frac{\lambda_2(x,y)}{\lambda_1(x,y)} \tag{58}$$

$$\theta(x,y) = \tan^{-1} \frac{((\lambda_1(x,y)^2) + (\lambda_2(x,y)^2))^{\frac{1}{2}}}{1 + L(x,y)} \tag{59}$$

Frame Level Max-Pooling Scale Selection:

We developed a novel frame level scale selection as described in Algorithm 8. First local shape (Eq. 58) and normalized curvature (Eq. 59) indices are computed at multiple scales. Sliding window template matching is performed between search window and tracked object template; and likelihood maps $\mathcal{L}_{\sigma_i}$ are obtained for each scale σ_i. Then, pixelwise maximum likelihood is computed as:

$$\mathcal{L}_{max}(x,y) = \max_{\sigma_i}(\mathcal{L}_{\sigma_i}(x,y)) \tag{60}$$

Finally, best scale σ^* is then selected as the scale that minimizes the mean square error between the pixelwise maximum likelihood $\mathcal{L}_{max}$ and each $\mathcal{L}_{\sigma_i}$.

$$\sigma^* = \underset{i}{argmin}(\sum_{x,y}(\mathcal{L}_{max} - \mathcal{L}_{\sigma_i})) \tag{61}$$

The described frame level max-pooling scale selection approach is applied to shape index and normalized curvature index separately. Figure 28 describes frame level max-pooling scale selection approach. Likelihood maps corresponding to selected scales are fused with likelihood maps for the remaining features. Proposed frame level scale selection ensures (i) scale coherence in the selected likelihood map (all pixels correspond to the same scale); and (ii) robustness against local outliers.

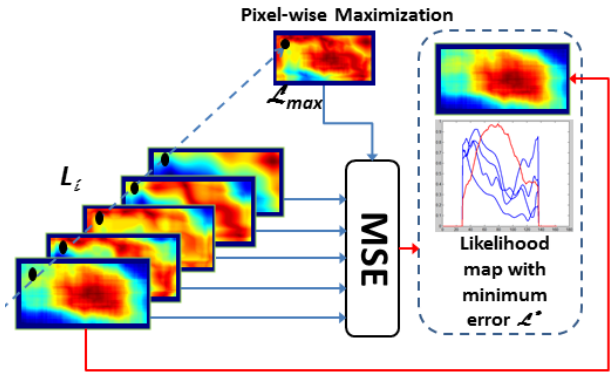


Figure 28: Frame level max-pooling scale selection approach.

For this project, we have developed both recognition-based single-object and association-based multi-object tracking systems. The goal of our single-object tracking systems is to track few objects of interest in the scene. The goal of our multi-object tracking system is to track *all* the moving objects in a particular region of interest for analysis of motion behavior patterns. MU single-object and multi-object tracking systems are briefly described below.

3.9.2 KC-LoFT: Kernelized Correlation Extended LoFT for Single Object Tracking

Correlation filter-based tracking is a discriminative method in which a filter based on the object appearance is used to estimate the most likely target location in the search window by distinguishing the target from its surrounding background. Convolution of the search window image with the filter is performed in Fourier

Algorithm 8 Frame level max-pooling scale selection

Input : I_{SW}, I_T: search window and object template
Output : $\sigma^*, \mathcal{L}^*$: best scale and corresponding likelihood map

1: **for** each scale i **do**
2: $L(x, y; \sigma_i) \leftarrow g(x, y; \sigma_i) * I(x, y)$
3: Compute Hessian matrix $H(x, y; \sigma)$ Eq. 56 and its eigenvalues λ_1,λ_2
4: Compute local shape features $\phi(x, y)$ Eq. 58, $\theta(x, y)$ Eq. 59
5: $\mathcal{L}_{\sigma_i} \leftarrow$ sliding window template matching between search window & object template
6: **end for**
7: Pixelwise maximum likelihood:
$\mathcal{L}_{max}(x, y) \leftarrow \underset{\sigma_i}{max}(\mathcal{L}_{\sigma_i}(x, y))$
8: Best frame level scale:
$\sigma^* \leftarrow \underset{i}{argmin}(\sum_{x,y}(\mathcal{L}_{max} - \mathcal{L}_{\sigma_i}))$

domain as element-wise multiplications to reduce complexity. The used filters are updated online to adapt to object appearance changes. Since their first use in visual tracking [40] [97], correlation filter-based tracking has been extended in multiple ways. [126][60] improved adaptation to scale changes; while [34] incorporated histogram based ridge regression learning to improve robustness to fast deformation and shape variations.

For a tracking system to handle a variety of challenging conditions, it is important to use multiple information cues robust to these conditions. Feature fusion needs to be explored in order to overcome weakness of individual information cues and to strength the overall process. Some feature fusion methods used in visual tracking are fixed linear combination [127] [121], adative weight update [56][80], variance ratio [56], Bayesian probability distrubuation[195], Bhattacharyya distance [12], peak-to-sidelobe ratio [40].

Our group developed a Likelihood of Features Tracking (LoFT) system[167] (described above) that fuses multiple sources of information about target and its environment to perform robust single object tracking. KC-LoFT extends and improves the original LoFT framework by incorporating: (i) a discriminative module using kernelized correlation filters to strength LoFT's adaptation to environment changes; (ii) a multi-dimensional kernelized template (beside LoFT's original multi-feature template) to ensure comprehensive localization of the target in different scenarios; and (iii) a decision module to adaptively update and fuse kernelized template. The new decision module uses peak-to-sidelobe ratio (PTSR) criterion to avoid fusing any irrelevant response from the kernelized correlation filters to the other LoFT feature maps and to prevent update of the regression parameters and correlation-based template during occlusion or other cases of sudden appearance change. We have tested the proposed KC-LoFT tracker on wide area motion imagery and compared its performance to non-deep learning trackers from VOT2015[116] and VOT2016[115] challenges (Table 4) whose codes were publicly available.

KC-LoFT kernelized correlation module uses two features, HoG and intensity, to localize the target within the search window. The two features are stacked to form a single vector x that is then used in the kernelized correlation filter (KCF) scheme. The goal of correlation filter is to minimize the square error over the sample x and the expected target y using the ridge regression loss function defined as follow

$$\min_{w} \sum_{i}^{n} (f(x_i) - y_i)^2 + \lambda \|w\| \tag{62}$$

where n is the length of the feature vector and λ controls the regularization parameter w over the linear combination of samples $f(x) = w^T x$. Following the description and derivatives on [97] regression learning parameter, α can be learned using

$$\hat{\alpha} = \frac{\hat{y}}{\hat{k}^{xx} + \lambda} \tag{63}$$

where $\hat{k}^{xx}$ is a correlation kernel. Gaussian kernel is used as follow

$$k^{xx'} = \exp(-\frac{1}{\sigma^2}(\|x\|^2 + \|x'\|^2) - 2F^{-1}(\hat{x} \odot \hat{x}'^*)) \tag{64}$$

Where $\hat{x}$ is the DFT of x, $\hat{x}'^*$ is the complex-conjugate of $\hat{x}'$, $\odot$ is an element wise multiplication, and $\hat{}$ is the discrete Fourier transform. To detect the position of the object, a new patch z (search window) is cropped form the location estimated from Kalman filter. The response is found according to the correlation between previously learned template $\tilde{x}$ and new patch z.

$$\hat{f}(z) = (\hat{k}^{\tilde{x}z})^* \odot \hat{\alpha} \tag{65}$$

The steps of the target detection and online training process are described in Algorithm 9, where θ is the learning rate assumed to be 0.1.

Algorithm 9 Target detection and training process using correlation filter module

Input : $P_{f-1}, z, \alpha_{f-1}, \tilde{x}$: predicted position, current patch (search region), dual space coefficient, the previous updated correlation-based template

Output : $P_f, \alpha_f, \tilde{x}_{new}$: estimated new position, the updated dual space coefficient, updated correlated-based template

1: Calculate the kernel $k^{\tilde{x}z} = \exp(-\frac{1}{\sigma^2}(\|\tilde{x}\|^2 + \|z\|^2) - 2F^{-1}(\hat{\tilde{x}} \odot \hat{z}^*))$
2: Calculate the response $\hat{f}(z) = (\hat{k}^{\tilde{x}z})^* \odot \hat{\alpha}_{f-1}$
3: Find the new position P_f from the maximum response $\hat{f}(z)$
4: Crop new patch x_{new} with P_f as a center of the region
5: Update the template $\tilde{x}_{new} = \theta\tilde{x} + (1-\theta)x_{new}$
6: Calculate $\hat{\alpha}_{new} = \frac{\hat{y}}{\hat{k}^{\tilde{x}x_{new}} + \lambda}$
7: Update dual space coefficient $\alpha_f = \theta\alpha_{f-1} + (1-\theta)\alpha_{new}$

3.9.2.1 KC-LoFT: Integration of Modules

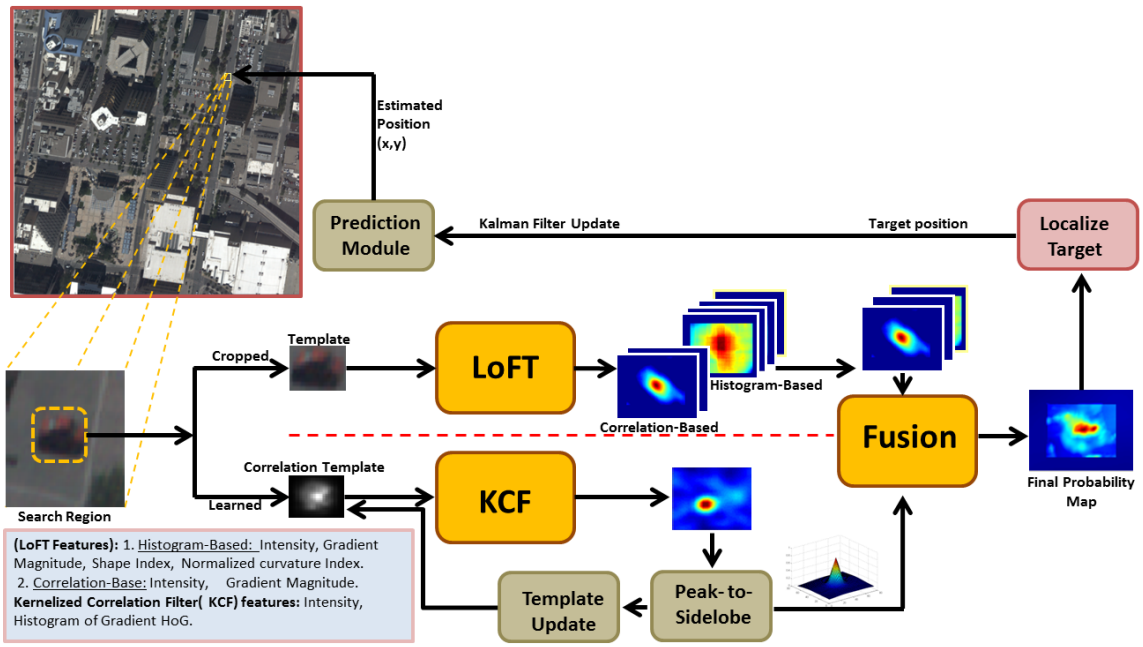


Figure 29: KC-LoFT template update and likelihood fusion pipelines.

KC-LoFT integrates the discriminative KCF module to the LoFT framework to enable robust and efficient localization of targets. KC-LoFT also incorporates an online template update scheme to LoFT

to prevent drifts and to enable robust handling of partial or full occlusions. Figure 24 illustrates modules involved in the proposed KC-LoFT pipeline.

Correlation filter learns filter parameters and updates template for each frame by including positive and negative examples within the search window using ridge regression. KC-LoFT fuses the responses from LoFT features with the correlation response from the correlation module to generate a final fused probability map that is used to localize the target. Because the correlation template update happens during the processing of every frame, online update on correlation-based template has faster response to appearance changes than appearance-based template. The response $\hat{f}(z)$ from the correlation filter module is fused with the other appearance-based template likelihood maps from LoFT to generate a final likelihood map to localize the expected position of the target. Peak-to-sidelobe ratio (PTSR) likelihood map evaluation criterion is used to avoid updating the correlation-based template and regression parameters during occlusion and to prevent correlation likelihood map to be fused with the remaining maps when it is unreliable. Peak-to-sidelobe ratio [40] evaluates the discrimination power of a likelihood peak as follow:

$$PTSR = \frac{P_{max} - \mu_{sidelobe}}{\sigma_{sidelobe}} \tag{66}$$

where P_{max} is the value of the maximum response, $\mu_{sidelobe}$ and $\sigma_{sidelobe}$ are the mean and standard deviation of the likelihood map values except an 11×11 area around the maximum peak. Figure 30 shows sample likelihood maps and associated PTSR measurements.

During occlusion events, PTSR helps suspend correlation template update until the object appears again. Figure 29 illustrates KC-LoFT template update and likelihood fusion processes. Suspension of the template update process is important to preserve the target template through occlusion events.

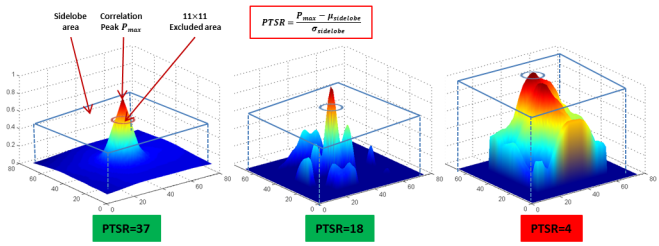


Figure 30: Sample likelihood maps and associated peak-to-sidelobe ratio values. Lower PTSR values indicate occlusions or other sudden appearance change scenarios and suspend the template update process.

3.9.3 MU Multi-Object Tracking

Multi-object tracking (MOT) is the process of locating objects of interest in a video sequence, maintaining their identity over time, and producing set of trajectories corresponding to their motion. Our MOT system is a tracking-by-detection framework consisting of three stages: (1) detection validation and filtering, (2) local data association, and (3) global data association (Figure 31). For each frame I_t of a given video sequence $V = \{I_1, I_2, ..., I_Q\}$ of length Q, there are set of $N(t)$ detected objects $D_t = \{d_{t,1}, d_{t,2}, ..., d_{t,N(t)}\}$ where $d_{t,i}$ represents object i in frame I_t. Each detected object $d_{t,i}$ is encoded with the vector $(d_{t,i}[x], d_{t,i}[y], d_{t,i}[w], d_{t,i}[h], s_{t,i})$, where the entries represent position, width, height, and detection score respectively. Tracking determines a set of associations $\mathbb{A} = \{\theta_2, \theta_3, ..., \theta_Q\}$, where each θ_t represents an assignment matrix from tracks to detections at frame I_t.

Detection filtering step is applied to eliminate potential false detections produced by the detector. Objects with low detection scores are removed to reduce false positive score. *Local data association* is an online step that links the detected objects in the current frame to the existing tracks. Hungarian Algorithm [148] is used to resolve associations on a cost matrix relying on spatial distances between tracks and detections. *Global data association* is an offline tracklet to tracklet (rather than object to tracklet) linking step. Tracklets (short tracks) terminated early because of problems such as occlusions or weak detection scores, are linked to tracklets formed after object recovery using object appearance cues and tracklet motion history. Appearance is described and compared using HoG descriptor [59] and CN (color name)

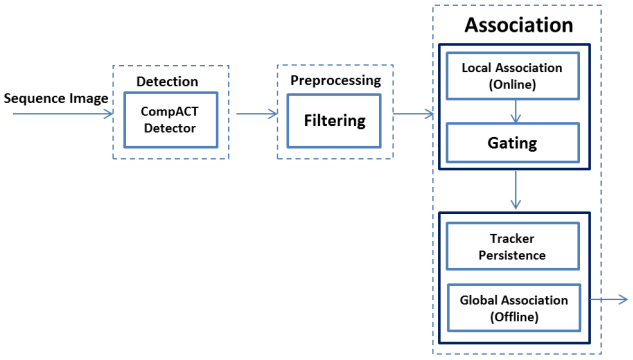


Figure 31: The framework of Multi-Object Tracking.

histogram along with a novel weighted CN histogram dissimilarity measure. Below sections summarize main steps involved in the proposed tracking system.

3.9.3.1 Track Initialization

When a new track is formed, three groups of information corresponding to the new track are initialized: (1) detected object's location, width, height, and appearance; (2) counters for age, visible frames, and invisible frames; and (3) Kalman filter parameters $KF_i^t = \{x_i^t, P_i^t\}$ where x_i^t represents state estimate for the object i at frame t and P_i represents associated covariance matrix.

3.9.3.2 Track Position Prediction

Association is done between detected object positions D_t and predicted track positions T_t at frame t. Constant velocity Kalman filter is used to predict track positions. The process involves two steps: (1) *prediction step* that uses the current state estimate $\{x_i^{t-1}, P_i^{t-1}\}$ from $t-1$ to predict the estimate for time t; and (2) *update step* where the current prediction $\{x_i^t, P_i^t\}$ is combined with current observation (detection) information $\{d_i^t, R_i^t\}$ to refine the state estimate for future predictions, where d_i^t is the position and R_i^t is the observation covariance matrix of the detected object i .

3.9.3.3 Local Data Association using Spatial Distance

For the tracking-by-detection systems, the biggest challenge is the association of the noisy object detections $D^t = \{d_1, d_2, ..., d_N\}$ in a video frame with the previously tracked objects $T^{t-1} = \{T_1, T_2, ..., T_M\}$. Detected objects are assigned to existing tracks by minimizing a cost matrix using Munkres Hungarian algorithm [148]. Elements of the cost matrix are computed as:

$$C(i,j) = \log \|d_i(x,y), T_j(x,y)\|_2 \tag{67}$$

where $d_i(x,y)$ and $T_j(x,y)$ are the centroids of detected objects and predicted tracks respectively. We use circular gating around predicted track positions to eliminate highly unlikely associations, to reduce computational cost, and to reduce false matches. Minimization on the cost matrix results in the assignment $T \times 2$ matrix $\mathbb{A}_t$ containing the indices of the corresponding detection and track pairs. $\mathbb{A}_t$ determines track states for frame t. The four possible states *{new track, extended track, lost track, inactive track}* are illustrated in Figure 32. State descriptions and associated parameter updates are summarized in Table 1.

3.9.3.4 Global Data Association using Spatial Distance and Appearance Similarity

Various problems during object detection and data association stages may cause tracks to terminate early resulting in short tracklets rather than full tracks. Global data association is used to link these tracklets to generate longer tracks. Global data association is an expensive process because it involves optimization of *all* possible matches not just the ones between consecutive frames. Reducing the number of objects that

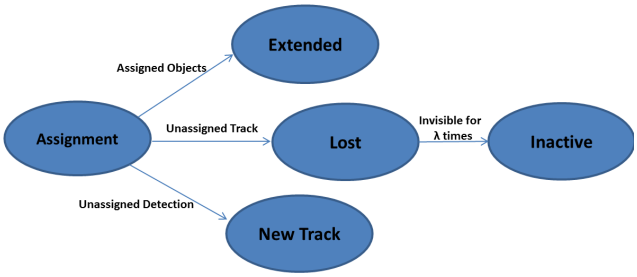


Figure 32: Decision State after assignment.

Table 1: State description and associated parameter updates.

State	Parameter Updates	Description
New Track	- Track ID - Kalman filter state $KF_i^t = \{x_i^t, P_i^t\}$ - Counters: age, visible frames, invisible frames - Appearance: CN color histogram & HoG descriptor	Detected objects not assigned to any existing tracks start new tracks.
Extended Track	- Kalman filter state $KF_i^t = \{x_i^t, P_i^t\}$ - Counters: age, visible frames	Detected objects successfully matched to existing tracks extend those tracks.
Lost Track	- Counters: age, invisible frames	Tracks not assigned to any detected objects are assigned to lost state.
Inactive Track	Save - Track ID - Full trajectory (previous and last seen positions) - Appearance: CN color histogram & HoG descriptor - Counters: age, born/death frames	After spending λ time steps in lost state, unmatched tracks are terminated.

need to be associated is helpful in reducing this computational cost. We use a refinement process relying on spatial distance, start and end frames, motion direction, and appearance model of the tracklets to filter out infeasible tracklet matches before the global assignment step. Given a tracklet K_i and the initial set of tracklets $J = \{K_1, K_2, ..., K_k\}$, the refinement process filters out the tracklets with the following properties from the set J as infeasible forward matches for K_i. Where forward linking refers to cases where tracklets are appended to the end of K_i.

1. All tracklets K_j that are initialized on frame borders or other source positions (tracklets entering the scene).
2. All tracklets K_j that are born before the death of tracklet K_i.
3. All tracklets K_j that start beyond the spatial distance T_s from the last position of tracklet K_i.
4. All tracklets K_j that start beyond the temporal distance T_t from the last frame of tracklet K_i.
5. All tracklets K_j whose appearance distances from tracklet K_i are above T_a. Appearance is described in terms of color using CN (color name) distribution and in terms of shape and texture using HoG descriptor.

Refinement process for backward linking where K_i gets inserted to the end of another tracklet is done in a similar way. Once all potential match sets are refined. Global data association determines the tracklet to tracklet associations by minimizing spatial and appearance distances. Details on our appearance model and appearance comparison scheme are given in Section 3.9.3.5.

3.9.3.5 Appearance Model for Tracklet Linking

It is important to have discriminant features to filter out infeasible detection-to-tracklet or tracklet-to-tracklet associations. Tracked objects' appearance models provide powerful information to refine the set of

candidate matches. However, appearance is also sensitive to external factors such as illumination changes, shadows, partial occlusions, pose, viewing angles etc. It is important to develop appearance models that can discriminate different tracked objects, while being invariant against factors such as illumination, pose etc. In this work, we propose an appearance model that combines shape and texture information using HoG descriptor with object color attributes using our novel color correlation cost matrix. HoG [59] is a widely used powerful descriptor that describes shape and texture through histogram of gradient orientations in local image regions. HoG is particularly suitable for description of rigid objects such as cars in the UA-DETRAC dataset [211]. We record the HoG descriptor for all new tracks at the time of track initialization. Mean square error (MSE) is used to compute the distance between HoG descriptors.

We incorporate color information through an extension of the CN (Color Names) model proposed in [206]. Linguistic study described in [161] shows that English language has eleven basic color terms: black, blue, brown, gray, green, orange, pink, purple, red, white and yellow. [206] explores this concept to generate CN (Color Names) map. CN model associates RGB color values with linguistic color labels and reduces the 256^3 RGB color space to just 11^1 CN color space. Biggest challenge in color description is sensitivity to illumination. Illumination variations and shadows may alter color (and intensity) of an object making the information not reliable (e.g. shadow may change blue to black, or yellow to brown). When performing color appearance comparison, it is important to consider similarity of individual color codes and likelihood of colors switching from one value to another (e.g. while blue, yellow, and orange are three distinct color names, distance between blue and yellow is higher than distance between yellow and orange, and transition likelihood from blue to yellow is lower compared to transition probability from yellow to orange). In [171], O'Pele et.al. proposed a color distance weight matrix fusing CIEDE2000 color dissimilarity [187] with CN color pair probabilistic distance matrix. In this work, we have built and used an 11×11 CN-to-CN color correlation weight matrix $\mathcal{W}_{CN}$ to account for similarity between different CN values during color distribution comparison. The elements of the color correlation matrix are computed as follows:

$$\mathcal{W}_{CN}(c_i, c_j) = 1 - 2 \times \underset{k}{max}\big(\min \big(\mathcal{G}(k,i), \mathcal{G}(k,j)\big)\big) \tag{68}$$

where c_i and c_j are two CN codes and $\mathcal{G}$ is the $2^{15} \times 11$ matrix describing the mapping from $32 \times 32 \times 32$ quantized RGB space to 11-valued CN space provided by [206]. In order to compare object color distributions, we use Earth mover distance (EMD) [182]. EMD computes the minimum paid cost to transfer one histogram to another histogram. We use color correlation matrix $\mathcal{W}_{CN}$ as transfer cost. See Figure 33 for more details. Use of cross-bin color histogram distance computation using EMD scheme and color correlation matrix $\mathcal{W}$ described above decreases sensitivity to color changes caused by illumination variations and improves tracking results compared to bin-to-bin color histogram comparison.

3.10 Feature Work

3.10.1 3D Surface Recovery, Regularization, Segmentation and Clustering

3.10.1.1 3-D Bilateral Filter for Point Cloud Smoothing

Various edge preserving denoising and smoothing techniques exist for removing noise and restoring faint structures in surface data. Variational and PDE based methods [28], bilateral filter [202], nonlocal means [42] are some of the powerful methods in edge preserving denoising. Extensions to 3D data which can be applied to surfaces, mesh and volumes have been considered [110, 75, 139]. The bilateral filter, in particular, is appealing for denoising 3D data due to its simple nature and its extensible properties [35]. The bilateral filter is an edge preserving denoising filter that has been widely used for image and image plus depth map smoothing. We briefly recall the bilateral filter (BLF) which is well-known in 2D image smoothing and sketch the mesh denoising version of it. The original BLF is of the form

$$I'(x) = K(x) \sum_{y \in N(x)} W_c(\|x - y\|)\, W_s(\|I(x) - I(y)\|)\, I(y), \tag{69}$$

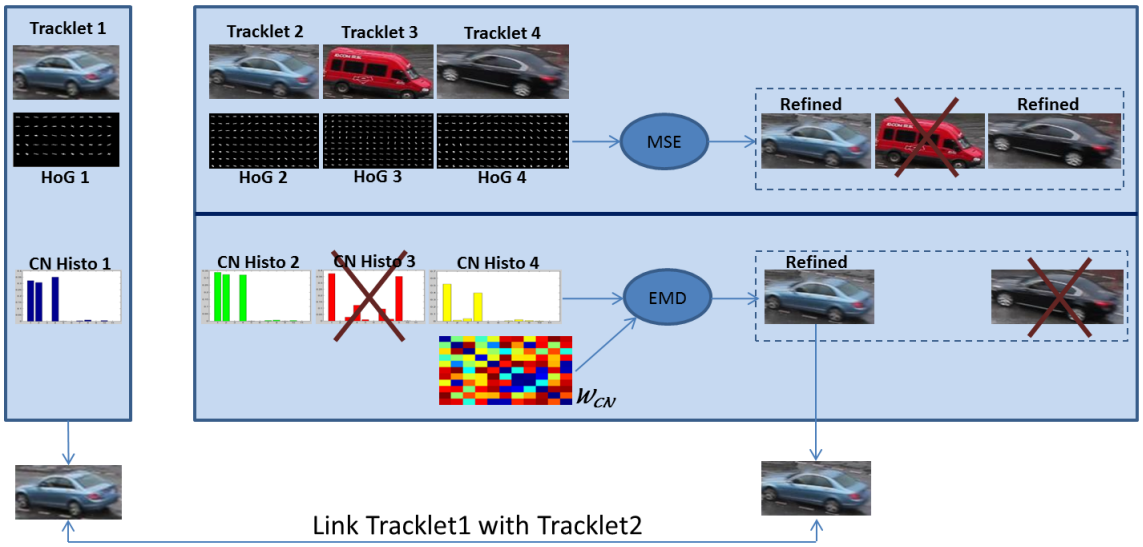


Figure 33: Apperance model refinment for global assignment. Tracklet_1 need to be assigned to one of the candidates ($\text{Tracklet}_{2,3,4}$) that born after Tracklet_1. In the first step, Tracklet_3 has been excluded by HoG constrain. Then, Tracklet_4 has been excluded by color constrain. Finally, After refinement Tracklet_1 is linked with Tracklet_2

where $N(x)$ defines the neighborhood of pixel position x, $K(x)$ the normalization factor, and $I(x)$ is the signal to be processed at x, W_c, W_s are usually chosen as Gaussian functions. To extend the BLF to mesh denoising, we need to apply the filter to the signal locally defined at each vertex. To obtain locally defined signal points we can either project a vertex to the first order surface predictors at its neighbors [110] or project its neighboring vertices to the predictor at the vertex [75].

3.10.1.2 Smoothed Signed Distance Surface Reconstruction

We consider several approaches to obtain a surface from a finite set of oriented points. One can utilize the three-dimension smoothed signed distance (SSD) function [47]. This method uses a variational formulation to reconstruct a watertight surface using implicit function and represents the surface as a signed distance field. SSD uses a Poisson reconstruction process and is defined in the continuous minimization form as:

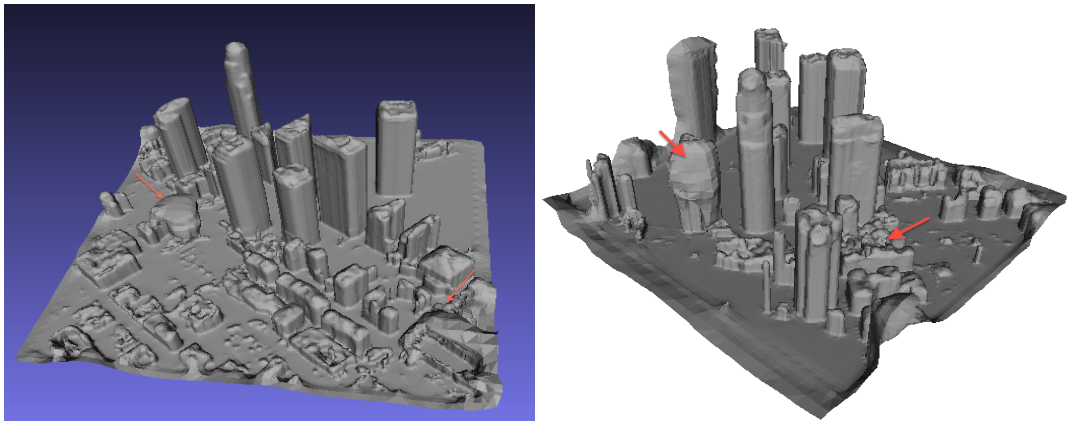
$$\min_f \frac{1}{N}\sum_{i=1}^{N} f(x_i)^2 + \frac{\lambda_1}{N}\sum_{i=1}^{N}\|\nabla f(x_i) - n_i\|^2 + \frac{\lambda_2}{|V|}\int_V \|Hf(x_i)\|\, dx, \tag{70}$$

where $\{(x_i, n_i)\}_{i=1}^N$ is the oriented point cloud (x_i surface location sample, n_i surface normal), $f : V \subset \mathbb{R}^3 \to \mathbb{R}$ is a signed distance field on a bounded volumetric domain V, H is the Hessian, λ_1, λ_2 constants, and N is the number of points in the data set. The resulting signed distance function $f(x)$ is then utilized to construct a mesh using dual marching cubes algorithm. The limitations of the schemes are extensive parameter tuning and depending upon the initial noise level in the point clouds the final SSD surfaces can have distorted structures, see Figure 34. Moreover, there is no explicit planarity model constraint and hence building tops can be distorted.

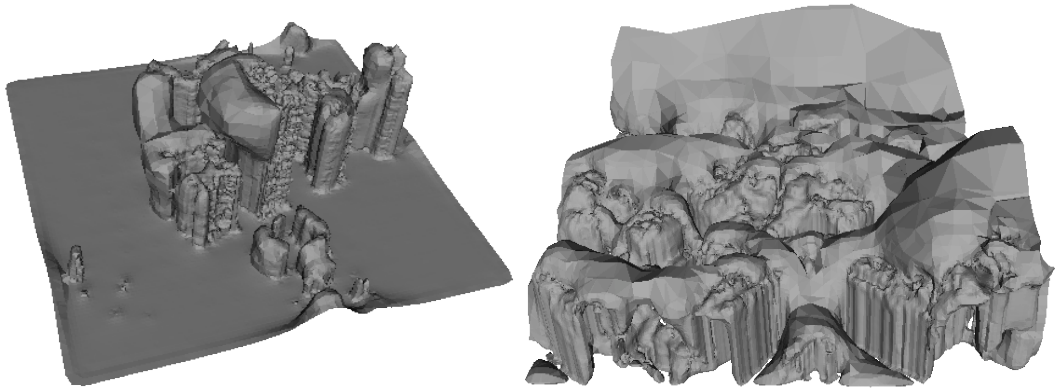
3.10.1.3 Variational Optical Flow Refinement of an Initial Point Cloud

Alternatively, assuming a rough surface S is obtained by either Delaunay triangulation or scattered interpolation of a sparse 3D feature point cloud, we can use dense correspondences computed using a fast and precise variational optical flow technique that extends the convex optimization approach [87].

Let K be the camera calibration matrix. Let $P_j = K[R_j|-R_j t_j]$ be the projection matrix corresponding to image I_j ($j = 1, 2, \ldots, N$ where N is the number of views used). Let $\mathbf{X}_i = (x_i, y_i, z_i)$ be the point on the surface S. Let $[u_i^j, v_i^j] = \mathbf{u}_i^j = \mathbf{P}^j(\mathbf{X}_i)$ be the projection of the (visible) point $\mathbf{X}_i$ on camera j. A



(a) Example SSD reconstruction with roof-tops distorted



(b) Partial failure

(c) Complete failure

Figure 34: Example surface reconstruction with the SSD method on Los Angeles point cloud data obtained using our voting based approach. Top row: Arrows indicate failure regions in SSD reconstruction. Bottom row: Failed reconstruction results due to sensitivity to noise in the initial point cloud.

small motion of the point $\mathbf{X}_i$ gives a new point

$$\tilde{\mathbf{X}}_i = \mathbf{X}_i + \triangle \mathbf{X}_i$$

The corresponding image displacement is given by,

$$\tilde{\mathbf{u}}_i^j = \mathbf{u}_i^j + \triangle \mathbf{u}_i^j$$

Note that the image displacement and surface displacement at a point $\mathbf{X}_i$ are related via the Jacobian:

$$\triangle \mathbf{u}_i^j = \mathbf{J}_{\mathbf{X}_i}^j \triangle \mathbf{X}_i$$

where

$$\mathbf{J}_{\mathbf{X}_i}^j = \frac{\partial \mathbf{P}^j}{\partial \mathbf{X}}\bigg|_{\mathbf{X}_i} = \begin{pmatrix} \frac{\partial \mathbf{P}_u^j}{\partial x} & \frac{\partial \mathbf{P}_u^j}{\partial y} & \frac{\partial \mathbf{P}_u^j}{\partial z} \\ \frac{\partial \mathbf{P}_v^j}{\partial x} & \frac{\partial \mathbf{P}_v^j}{\partial y} & \frac{\partial \mathbf{P}_v^j}{\partial z} \end{pmatrix}\bigg|_{\mathbf{X}_i}$$

The following main steps are involved in estimating the $\tilde{\mathbf{X}}_i^k$ at a fixed camera image k:

- Fix a camera k, then all points must satisfy:

$$\triangle \mathbf{X}_i^k = \lambda_i \mathbf{r}_i^k$$

where $\mathbf{r}_i$ is the unit vector direction at camera k. Then for all the cameras j 'nearby',

$$\triangle \mathbf{u}_i^j = \mathbf{J}_{\mathbf{X}_i}^j \triangle \mathbf{X}_i = \mathbf{J}_{\mathbf{X}_i}^j \lambda_i \mathbf{r}_i$$

- Compute $\triangle \mathbf{u}_i^j = \mathbf{u}_i^j - \tilde{\mathbf{u}}_i^j$:
  - First project $\mathbf{u}_i^k$ onto the base surface S to get $\mathbf{X}_i^k$, project this onto nearby cameras j to get $\mathbf{u}_i^j$.
  - $\tilde{\mathbf{u}}_i^j$ is given by the optical flow computed at $\mathbf{u}_i^k$.
- Solve the minimization:
$$\min_{\lambda_i} \left\| \mathbf{J}^j_{\mathbf{X}_i} \mathbf{r}_i \lambda_i - \triangle \mathbf{u}_i \right\|$$
using the closed form solution
$$\lambda_i = ((\mathbf{J}^j_{\mathbf{X}_i} \mathbf{r}_i)^T \mathbf{J}^j_{\mathbf{X}_i} \mathbf{r}_i)^{-1} \mathbf{J}^j_{\mathbf{X}_i} \mathbf{r}_i \triangle \mathbf{u}_i$$
- Finally,
$$\tilde{\mathbf{X}}_i^k = \mathbf{X}_i^k + \mathbf{r}_i \lambda_i$$

3.10.1.4 Simultaneous Smoothing and Depth Map Fusion Using Variational Methods

We propose to utilize regularization methods for smoothing the noisy point cloud surfaces without introducing distortions or removing salient 3D structures. In particular we focus on total variation (TV) based methods which is a class of well-known edge preserving regularization approaches [87]. If we let $\mathbf{u} : \Omega \subset \mathbb{R}^3 \rightarrow [-11]$ be the truncated signed distance function, then the following minimization can be used to obtain simultaneous smoothing and fusion of depth-map.

$$\min_{\mathbf{u}} \int_\Omega |\nabla \mathbf{u}| + \lambda \sum_{i=1}^{N} \int_\Omega h(\mathbf{x}, i) \left| \mathbf{u}(\mathbf{x}) - d_i \right| \, d\mathbf{x} \tag{71}$$

where $h(\mathbf{x}, i)$ is the histogram count of bin i at voxel $\mathbf{x}$, N the number of depth maps and d_i distance for histogram bin i. The advantages of the above variational method are (a) efficient convex optimization based solvers which can handle millions of points (b) data-driven approach to handle noisy point clouds and (c) extensible for further optical flow based refinements.

To obtain highly accurate large-scale dense 3D reconstruction variational - partial differential equation (PDE) approaches can be utilized [24, 117]. In particular, we will consider variational regularization methods based on total variation (TV) and total generalized variation (TGV) type functionals to be used along with our 3D reconstruction pipeline for improved discontinuity preservation. For example to compute the depth-map Z between two images I_l and I_r a general variational method can be posed as an energy minimization of the form:

$$\min_Z \mathcal{E}(Z) = \int_\Omega (I_1(x) - I_r(f(x, Z(x))))^2 dx + \nu \int_\Omega \phi(|\nabla Z|) dx \tag{72}$$

where $\phi(\cdot)$ is the regularization function which can be designed to prevent smoothing of discontinuities allowing sharp depth map edges to be preserved and the function $f(\cdot, \cdot)$ depends on the camera parameters. We plan to extend this general approach for two or more images as well as investigate various regularization functions which can aid in improving final reconstruction quality and maintaining fine geometric details. Further, we plan to leverage state of the art convex optimization methods for efficiently solving these minimization problems and use the fastest numerical algorithms for the regularization part of the 3D reconstruction process.

3.10.2 GPS Denied Environments

In GPS denied environments only partial camera pose metadata will be available; namely the latitude and longitude location information of the camera and platform will not be available or only sporadically available. We will investigate a few approaches for bundle adjustment algorithms with partial information in

the context of Simultaneous Localization and Mapping (SLAM). SLAM is a core requirement to enable navigation, targeting and mission execution in adverse A2AD (anti-access or access denied) environments. SLAM is a fundamental requirement in mobile robotics for the autonomous platform to be aware of its environment, and deals with building a geospatial map of the environment while simultaneously determining the pose of the robot or platform with respect to the same map. The assumption is that the environment has a set of stationary features or landmarks which can be observed by sensors from the robotic platform. For GPS-denied SLAM we propose to use a sparse local bundle adjustment approach using orientation only noisy IMU sensor measurements, a partial a priori or sensed 3D scene model (that could be stored on-board) and an incremental light bundle adjustment method like that proposed in [103].

Over the past few decades there has been extensive effort in the robotics, computer vision and photogrammetric communities in finding a robust solution for SLAM due to a variety of compelling applications including autonomous (air, ground, sea, underwater) vehicles for defense requirements, hazardous environments, search and rescue, transportation (ie self-driving vehicles), entertainment, etc. Many current techniques for SLAM are derived from recursive probabilistic Bayes methods the most popular of which is the Kalman Filter (KF) in which posteriors are represented by Gaussians (Se, 2002). There are several variations of KF to deal with non-linear system dynamics such as the Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF), Covariance Intersection, Information Filtering etc. Another technique used in the FastSLAM implementation is Particle Filtering (PF) which is a recursive Bayesian Þlter with Monte Carlo sampling of the estimated probability density function [142]. PF implements Sequential Monte-Carlo (SMC) estimation using a set of random point clusters (*particles*) representing the Bayesian posterior. Expectation Maximization (EM) is another method used in SLAM. It is a statistical algorithm based on Maximum Likelihood (ML) estimation which is ideal for map construction but not for localization and is normally used when the robot pose is known within a given map. The most likely map estimate is then updated given the pose and initial map. As the platform moves, EM builds the map while increasingly improving the result. EM is normally used when the mapping problem is decoupled from localization. In such a situation it can be combined with a filtering method such as PF for localization. EM is advantageous to KF since it handles the data association or correspondence problem.

Full SLAM: The current state-of-the-art approaches are based on pose graphs using Dynamic Bayesian Networks and global bundle adjustment (BA) [204] combining vision-based electro-optical, range-bearing, bearing-only or range-only sensor measurements. Full-SLAM for autonomous robotic systems is feasible if the history of observations is maintained for a chunk of time to enable global optimization at the cost of additional computational resources. Full-SLAM refers to solving a non-linear optimization problem using the entire history of observations to provide a SLAM estimate for the autonomous platform rather than only the current state; it produces better results than filtering approaches and can handle non-Gaussian noise processes. In full-SLAM, the platform trajectory and map structure are jointly optimized given all measurements up to the current time. It is analogous to the BA approach widely used in the computer vision community for multiview 3D reconstruction. BA (full-SLAM) is an improved version of the Gauss-Newton Method using the Levenberg-Marquardt (LM) approach to optimize the parameters given some constraints such as image projection errors of 3D scene points [107]. Compared to KF-based SLAM, BA provides more accurate results, but speed and computational complexity are issues in using full-SLAM for realtime applications. Recent approaches to improve performance include the key-frame technique in adaptive relative bundle adjustment to increase the speed of full-SLAM [191]. In [191, 146] a method called Local Bundle Adjustment (LBA) is proposed which is able to perform BA over partial clusters of cameras using an intelligent method of choosing key-frames based on the estimated measurement uncertainties. Relative Bundle Adjustment (RBA) is another approach [191]. FrameSLAM [113] was proposed to match visual frames with large numbers of point features, using classic BA techniques, but keep only relative frame pose information (a skeleton). The skeleton is a reduced nonlinear system that is a faithful approximation of the larger system, and can be used to solve large loop closures quickly, as well as forming a backbone for data association and local registration.

GPS-denied SLAM: SLAM in GPS-denied situations for aerial platforms in unknown environments without a priori information is required for robust autonomous operation and mission success. GPS-denied SLAM has application for operations in dense, urban environments with weak GPS signals, interference,

conserving power, etc. [53]. In the absence of GPS, vision-based electro-optical and infrared cameras and IMU sensors can be used for UAVs [98]. In [123], visual and inertial information are fused using a non-linear optimization approach. They integrate an IMU error term with landmark reprojection errors in a probabilistic manner, resulting in a joint non-linear cost function that is optimized. Using the concept of *key-frames*, old states in the system are marginalized to maintain a bounded-sized optimization window, ensuring real-time operation. We will develop incremental localization estimation using local bundle adjustment based on a small cluster of views and using a priori (partial or complete) 3D models or landmarks that have been measured beforehand.

3.10.3 Alternative Flightpath Trajectories

In addition to circular orbits for which the current 3D pipeline and suite of algorithms has been developed, we will also evaluate the feasibility of 3D reconstruction under different sampling conditions as well as more general non-circular flightpaths. Such flexibility is essential for several reasons. One is wider coverage over a shorter period of time for large area mapping. Fully circular orbits are expensive in terms of platform time and require additional time to maneuver to the next circular flightpath for coverage of adjacent geospatial regions on the ground. Furthermore, circular flightpaths do not offer flexibility under hazardous adversarial situations. We will look into the feasibility of using several flightpath scanning patterns for 3D reconstruction including grid patterns, snake-like zig zag back and forth scanning with simultaneous gimbal tracking for several ground points. The second is to identify the technology gaps for 3D reconstruction in A2AD environments where airspace is not accessible or hazardous.

Dataset spec.			BA4S		BA4S					VisualSfM			Mavmap			Pix4D		
												Speed-up			Speed-up			Speed-up
Dataset	Image Size	n	n_o	m	Feat.+Track	Triangulation	Optimization	Total	Per Image	Total	Per Image	*BA4S / VSFM*	Total	Per Image	*BA4S / Mavmap*	Total	Per Image	*BA4S / Pix4D*
DinoRing	640×480	48	7K	2K	5s	∼0s	0.5s	0m+6s	0.12s	16s	0.33s	1.8×	*NA*	*NA*	*NA*	*NA*	*NA*	*NA*
Fountain-P11	3072×2048	11	52K	17K	3s	∼0s	2s	0m+05s	0.45s	18s	1.64s	3.64×	*NA*	*NA*	*NA*	*NA*	*NA*	*NA*
Four Hills, NM	6048×4032	100	263K	81K	42s	∼0s	4s	0m + 46s	0.46s	36m+00s	21.60s	47.0×	34m	20.04s	43.6×	41m	24.7s	53.7×
PIXHAWK[184]	752×480	300	254K	46K	10s	∼0s	18s	28s	0.09s	*NA*	*NA*	*NA*	19m	3.80s	42.2×	*NA*	*NA*	*NA*
Columbia, MO	6600×4400	202	656K	116K	87s	∼0s	14s	1m + 41s	0.50s	*NA*	*NA*	*NA*	60m	17.82s	35.6×	3h + 36m[a]	64.2s	128.3×
ABQ215, NM	6600×4400	215	668K	142K	93s	∼0s	24s	1m + 57s	0.54s	4h + 25m	73.95s	107.8×	1h + 07m	18.70s	34.6×	3h + 32m	59.2s	109.6×
Berkeley, CA	6600×4400	220	683K	139K	93s	∼0s	15s	1m + 48s	0.49s	4h + 40m	76.37s	155.9×	1h + 03m	17.80s	36.3×	3h + 52m	63.3s	129.2×
Los Angeles, CA	6600×4400	351	1,1M	207K	141s	∼0s	40s	3m + 01s	0.51s	8h + 05m	78.29s	153.5×	1h + 43m	17.78s	34.9×	8h + 55m[a]	91.45s	179.3×
ABQ1071, NM	6600×4400	1,071	3.5M	603K	440s	9s	251s	11m + 40s	0.65s	26h + 36m	89.41s	137.6×	5h + 21m	17.98s	27.7×	50h + 49m[b]	170.8s	262.8×
Columbia-II*, MO	6600×4400	5,322	17.4M	2.5M	2089s	67s	349s	41m + 45s	0.47s	*NA*[b]	*NA*	*NA*	*NA*	*NA*	*NA*	*NA*[b]	*NA*	*NA*
Columbia-II, MO	6600×4400	5,322	17.4M	2.5M	2307s	68s	916s	54m + 51s	0.62s	*NA*[b]	*NA*	*NA*	*NA*	*NA*	*NA*	*NA*[b]	*NA*	*NA*
Col 1	Col 2	Col 3	Col 4	Col 5	Col 6	Col 7	Col 8	Col 9	Col 10	Col 11	Col 12	Col 13	Col 14	Col 15	Col 16	Col 17	Col 18	Col 19

(a: Failure in its first attempt and had to be repeated. b: Failed to generate result.)

Table 2: Dataset characteristics and timings for individual processing steps (per image) and the full BA4S pipeline. We also include timing comparison with VisualSfM [217] and two aerial imagery SfM algorithms, Mavmap[184] and Pix4D[7]. Columns n (Col 3), n_o (Col 4) and m (Col 5) indicate number of cameras/images, number of 2D observation points and number of 3D points (feature tracks), respectively. The time for each stage of BA4S processing including feature extraction and tracking, triangulation and optimization is shown. All of the experiments in Table 2 include optimization for extrinsic parameters (rotation and translation) of every single camera ($6 \times n$ parameters), all 3D points ($3 \times m$ parameters) and one shared focal length. Only "Columbia-II*" does not include focal length optimization. The total time taken for BA4S, VisualSFM, Mavmap and Pix4D are shown along with per image timings. On average BA4S (on WAMI datasets) is nearly 130 times faster than VisualSfM (Col 13), over 35 times faster than Mavmap (Col 16) and about 274 times faster than Pix4D (Col 19).

4 Results and discussion

4.1 BA4S

In this section our developed BA4S SfM pipeline is evaluated. We provide a set of quantitative and qualitative assessments to evaluate the accuracy, speed and effectiveness of the proposed approach.

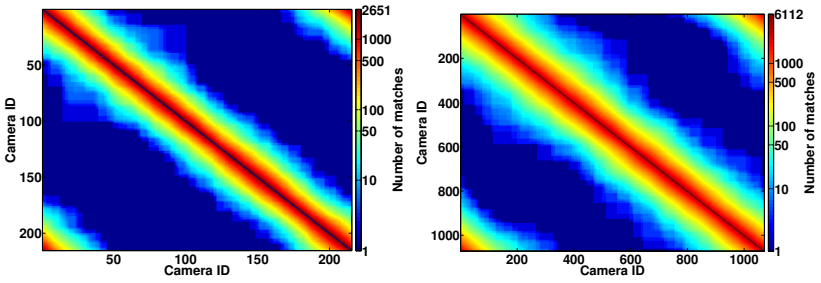
4.1.1 Implementation

The BA4S pipeline was implemented in C++. The computer used for experiments was a server with CPU Intel Xeon 5650, 2.66 GHz, 12 cores, 24 GB RAM and nVidia GTX480/1.5GB GPU. SIFT-GPU [214] was used for fast feature extraction. The Ceres Solver library [9] was used for non-linear least-squares estimation; Schur's complement, Cholesky factorization and Levenberg-Marquardt algorithms were used for trust region step computation.

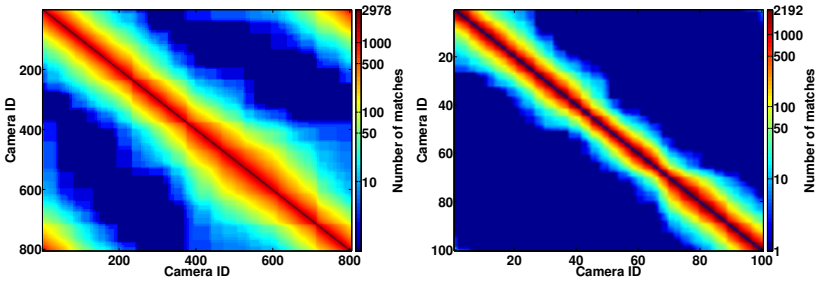
4.1.2 Datasets

We used a collection of real aerial WAMI datasets, most of them collected (by Transparent Sky [5]) using a fixed wing aircraft with on-board pose sensors flying over several different urban areas including downtown Albuquerque, NM, Four Hills, NM, Columbia, MO, Los Angeles, CA and Berkeley, CA. Fig. 38 shows a few sample frames from some of these datasets. The characteristics of the sequential and WAMI test datasets along with timing results are given in Table 2. Each dataset includes platform-based camera orientation matrices and translation vectors provided by imprecise IMU and GPS measurements along with intrinsic camera parameters that we refer to as metadata.

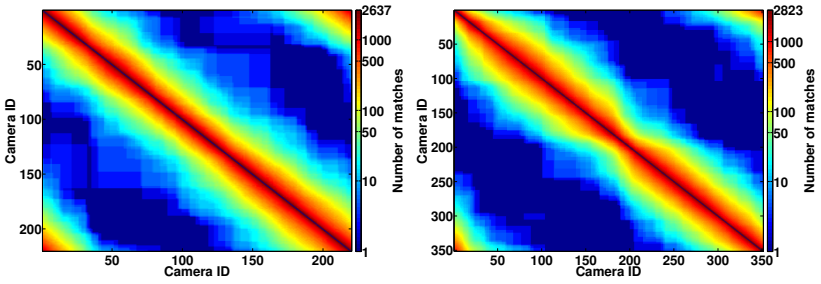
In addition to sequential aerial imagery datasets, the BA4S pipeline has been tested on several publicly available vision benchmark datasets with multiview imagery acquired in a sequential (circular) trajectory, including *DinoRing* from the Middlebury MVS evaluation project [186]. As discussed in [82, 13], *DinoRing* is a challenging dataset for evaluating SfM pipelines since the object lacks salient features making feature tracking very difficult. Another sequential multiview dataset is *Fountain-P11* from EPFL [199] with eleven images taken from different side views of a wall-attached fountain.



(a) ABQ215 (b) ABQ1071

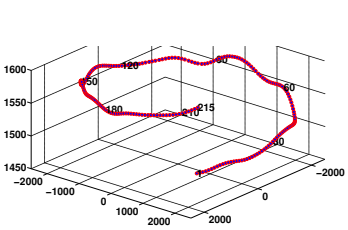


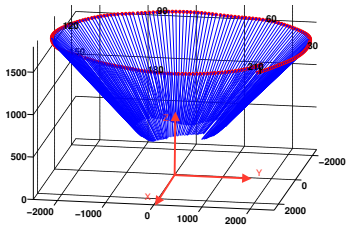
(c) Columbia (d) Four Hills



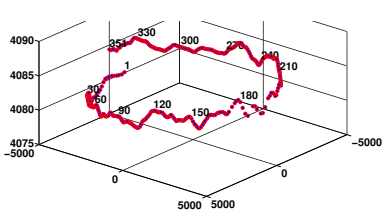
(e) Berkeley (f) LA

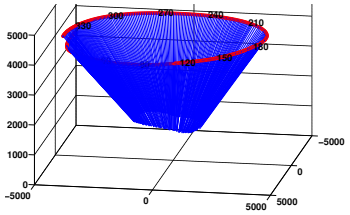
Figure 35: Plots for number of matches between camera pairs within the WAMI datasets. As expected from a sequential imagery, adjacent cameras share the largest number of matches; the number of matches decreases as the temporal distance between frames increases. The peak values are along the diagonals. One can see that there are large numbers of matches between the first frames and last ones in most of these WAMI datasets, as a loop closure was performed.





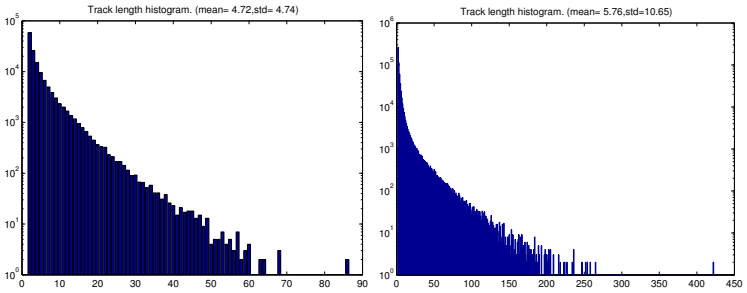
(a) ABQ215– Left: GPS positions. Right: GPS positions and IMU directions.



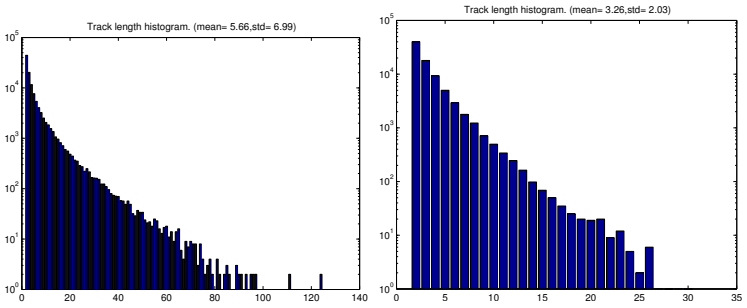


(b) LA– Left: GPS positions. Right: GPS positions and IMU directions.

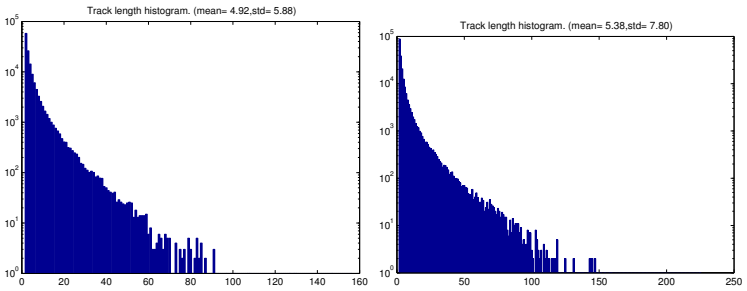
Figure 36: Uncorrected camera trajectories prior to applying BA4S corresponding to the Albuquerque (ABQ215) and Los Angeles (LA) aerial WAMI datasets. Note that the vertical axis of each graph uses a different vertical scale for the altitude range.



(a) ABQ. 215 frames; 141,559 tracks (b) AlbuquerqueFull. 1071 frames; 603,119 tracks



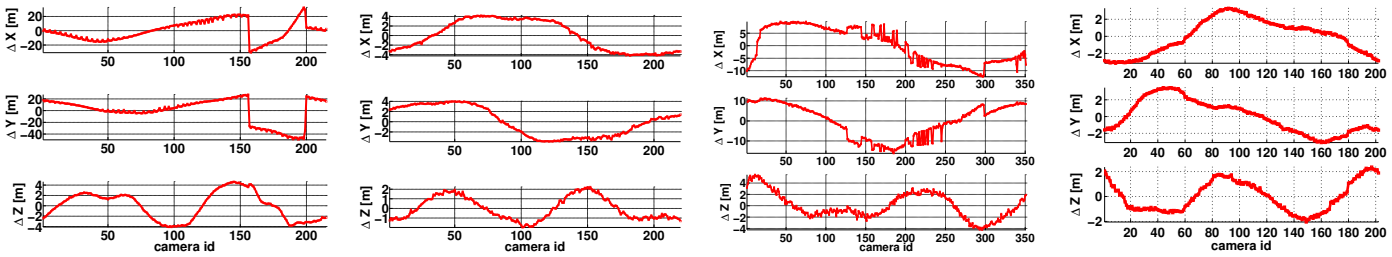
(c) Columbia. 202 frames; 115,897 tracks (d) Four Hills. 100 frames; 80,661 tracks



(e) Berkeley. 220 frames; 138,743 tracks (f) LA. 351 frames; 207,391 tracks

Figure 37: Distribution of feature track lengths in the WAMI datasets follows an approximately power law distribution.

Figure 38: Each row shows a different WAMI sequence collected by Transparent Sky [5] along with four frames well separated in time. Subsampled (but uncropped) frames from the original 6600×4400 images are shown except Four Hills which has a frame size of 6048×4032. See Table 2 for details of each dataset specification.



(a) ABQ215 (b) Berkeley (c) LA (d) Columbia

Figure 39: Differences in camera positions between original noisy metadata and BA4S corrected position output for different aerial WAMI datasets including ABQ, Berkeley, LA and Columbia. The curves show the amount of correction applied to each camera position after BA4S.

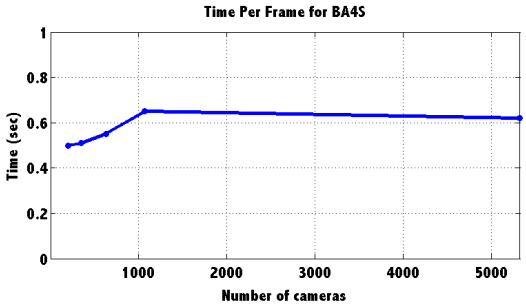
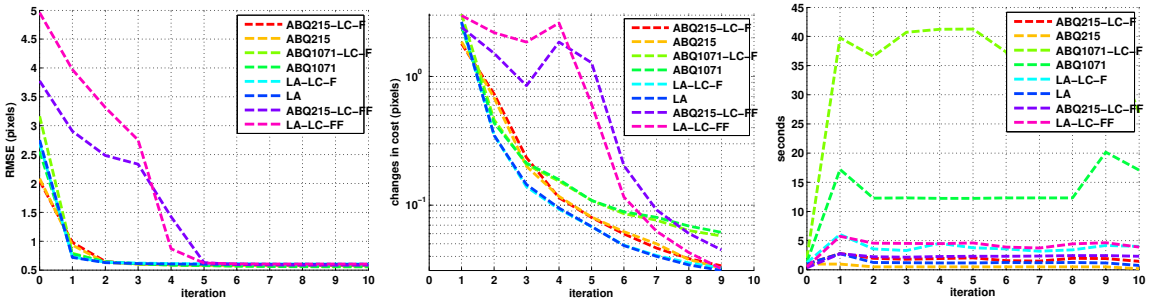


Figure 40: Per-frame timing performance for BA4S corresponding to the results in Table 2. The time per frame is approximately constant for 1000 of more cameras/views which is very promising for large scale SfM applications.

4.1.3 Results

We ran the proposed BA4S pipeline on each dataset in Table 2. The introduced sequential feature tracking method was used to track the identified SIFT features over the sequence (see Table 2 for the statistics). The histograms of the track lengths for six WAMI aerial dataset are presented in Fig. 37. Plots in Fig. 35 show the amount of matches (feature point correspondences) between each two images/cameras for the same six WAMI datasets. Notice that the feature matching and track building were performed sequentially and the last frame in each dataset was matched against the first one to avoid drift in the sequences (in robotics it is known as *loop closure* [79]). A linear triangulation algorithm [95] was used to estimate and initialize 3D points. The persistency factors of the tracks and their related statistics were used as weights in the adaptive robust function for the BA optimization. Fig. 36 depicts the camera positions and their viewing directions (optical axes) for two of the datasets, ABQ215 and LA. Fig. 39 shows the amount of camera position correction estimated by BA4S for three different WAMI datasets. The sensor measurement errors along the x- and y-axes directions for ABQ have the largest range while the Berkeley raw measurements were the most accurate. Notice that since the distance from the sensor to the scene objects is on the order of a kilometer or more, even small errors in the camera pose could lead to very large errors once projected onto the scene. For all the experiments presented in Table 2, the optimizations were performed on 6 extrinsic parameters (3 rotation and 3 translation) for every single camera ($6n$), all 3D points ($3m$) and one shared focal length (total number of parameters to be optimized were $6n + 3m + 1$). Only for "Columbia-II*", the focal length was not considered for the optimization. We observed that when the focal length is not included as a parameter to be optimized, we gain about $\%30$ speed up in the BA. This can be seen in Table 2 by comparing the timings in the "Optimization" column (**Col 8**) between Columbia-II*" and "Columbia-II". BA4S performance in terms of per frame execution times for WAMI aerial datasets is plotted in Fig. 40. The time per frame is approximately constant (see Column 10 in Table 2) and is independent of the number of cameras (or views) which is a surprising result compared to other SfM methods in the literature and also very promising for large scale aerial imagery BA, mosaicing, georegistration and 3D reconstruction applications. The timing for BA4S includes I/O for copying image frames from storage to memory which we found to be a significant percentage of the Feature Extraction and Tracking time (Column 6 in Table 2).

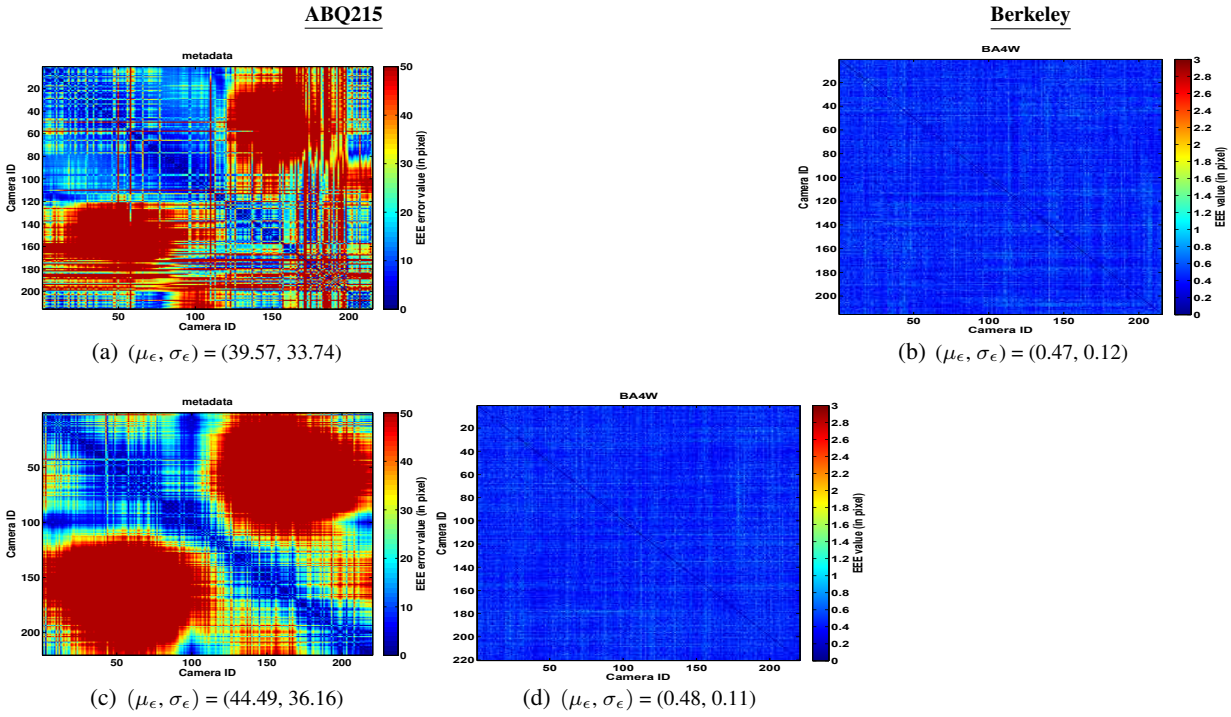


(a) RMSE (b) Change in RMSE per iteration (convergence) (c) Time per iteration

Figure 41: Reprojection error (RMSE) and convergence in BA4S. Legend notation: The first term in the legend is the dataset name. 'LC' means that a 'loop closure' was applied. 'F' is for the case with shared focal length of the cameras that is optimized using an initial value of 17,000 pixels which is close to the true value of 17,651 pixels. 'FF' is similar to the 'F' case, but using an initial value of 10,000 which is considerably far from the actual value (more than 50% deviation). As shown in the left plot, BA4S managed to recover and reduce the RMSE error to less than one pixel, even when the focal length was initialized to almost half of its actual value.

Convergence: Plots in Fig. 41 show the reprojection error and convergence of BA4S on a couple of datasets. The reprojection errors (RMSE) are shown in Fig. 41-left. The convergence plot or the amount of change in RMSE between each iteration is plotted in Fig. 41-middle and the consumed time per iteration is shown in Fig. 41-right. In some of these experiments a loop closure was performed (indicated by 'LC' in the captions). In this part of our evaluation (plots of Fig. 41), all of the experiments include optimization for extrinsic parameters (rotation and translation) of every single camera ($6 \times n$ parameters) and all 3D points ($3 \times m$ parameters). In some of these experiments the camera's focal length (as an intrinsic parameter) was also optimized where the initial value was set to 17,000 pixels, while the correct focal length was 17,650 pixels. Those experiments are indicated by 'F' in their captions in Fig. 41. In most of the cases, RMSE was significantly reduced to less than a pixel right after the first iteration in the non-linear optimization. For two datasets the focal length was set to 10,000 pixels which is far away from its actual value of 17,650 pixels (about 57% off). They are indicated by 'FF' in their captions. As one can see in Fig. 41-left, even when the focal length was initialized to almost half of its correct value, BA4S managed to recover and reduce the RMSE error to less than one pixel in its fifth iteration. This indicates that BA4S is not sensitive to the initial focal length and is able to recover in a few iterations. Consequently, the need for applying a camera calibration procedure (before BA4S) is relaxed. We compared the results of our BA4S with parallel VisualSfM/Bundler [217], a state-of-the-art SfM implementation, for some sample datasets; For VisualSfM we provided imagery without including metadata, which is not a fully supported feature. VisualSFM uses two strategies for matching including preemptive and exhaustive [215]. With preemptive matching VisualSFM generated several fragments of cameras and only a fraction of cameras could be recovered while for other cameras it failed. This is consistent with similar observations about VisualSFM's limited performance on sequential aerial images [184]. We ran VisualSFM in its exhaustive and expensive matching mode in order to recover all cameras. We compared our algorithm versus Mavmap[184] as a recent SfM algorithm for sequential aerial images. Like our method, Mavmap also takes advantage of temporal consistency and availability of metadata. The camera poses in Mavmap are initialized by estimating the essential matrix [152] and applying a RANSAC technique to deal with large amounts of outliers. In addition, we evaluated our method against Pix4D [7] which is a commercial SfM and aerial mapping system. In these experiments, our BA4S algorithm is (on WAMI aerial imagery) in average 130 times faster than VisualSfM, 35 times faster than Mavmap, and 274 times faster than Pix4D.

We also tried MapTK [122], a recent open-source SfM algorithm designed specifically for aerial imagery, on some of our WAMI datasets including ABQ215, Berkeley and ABQ1071. Its feature tracking (combined with parameter estimation and triangulation) part ran on all of these three datasets, however for Berkeley and ABQ1071 its optimization algorithm did not terminate after entering the BA part and we had to stop it after running for a long time. In its successful run on ABQ215, it took about 22m+14s for the feature extraction, parameter estimation and triangulation and 7m+17s for the optimization (in total,



(a) $(\mu_\epsilon, \sigma_\epsilon) = (39.57, 33.74)$ (b) $(\mu_\epsilon, \sigma_\epsilon) = (0.47, 0.12)$

(c) $(\mu_\epsilon, \sigma_\epsilon) = (44.49, 36.16)$ (d) $(\mu_\epsilon, \sigma_\epsilon) = (0.48, 0.11)$

Figure 42: Evaluation of camera parameters using EEE measure before (a and c) and after BA (b and d) for the ABQ215 (a and b) and Berkeley (c and d) aerial WAMI datasets. The pel in each matrix, ϵ_{ql}, indicates the error between the q-th camera (matrix rows) and l-th camera (columns), computed using (35). The ϵ_{ql} pel values were truncated to the maximum values 50 and 5 respectively. The mean and standard deviation of the errors in pixels (36), over all cameras are shown below each plot. Notice that each plot uses a different scale for better visualization of errors. Color bars shown to the right.

29m+31s) compared to our BA4S which took 1m+33s for feature extraction and tracking, and 24s for BA optimization (in total, 1m+57s) on the same dataset. As one can see, despite the fact that it is very recent and specifically designed for aerial imagery, and while being the fastest SfM (compared to other existing approaches), MapTK is still 15x slower than our proposed BA4S on the only dataset for which it was able to run without failing. We believe that MapTK like other SfM approaches is not able to truly use the available metadata (although noisy) towards achieving higher speed and robustness. It is due to the fact that the pipeline still uses RANSAC and iterative random based parameter estimations (see the conventional pipeline in Fig. 3.)

The EEE evaluation metric previously explained was applied to both the ABQ215 and Berkeley datasets and the symmetric error matrix ϵ_{lm} in (35) is shown as colored pels in Fig. 42. Plots (a) and (c) show the EEE measure of the camera parameters using metadata (uncorrected platform camera parameters); the range of errors is truncated to 50 pixels. The initial raw metadata is very noisy but after refinement using the proposed BA4S pipeline there is significant improvement in quality as shown in plots (b) and (d); notice the range of errors is truncated to 3 pixels in this case. The EEE μ_ϵ and σ_ϵ statistics using all the cameras (see Equation (36)) are also given. BA4S was quite successful in refining the metadata while using significantly less time (Table 2). For MapTK, the EEE (error) pairwise plot and also its histogram are shown in Fig. 43. The mean and standard deviation of EEE corresponding to this experiment are 2.31 and 2 pixels, respectively.

Fig. 44 shows the EEE graphically to assess camera parameters for ABQ215 (first two rows) and Berkeley (last row) datasets. Point correspondence #2 in the ground-truth between the 50th and 150th cameras within the sequence were used. The epipolar lines corresponding to image #50 in each dataset is computed using the camera parameters and plotted on image #150. The left column shows original raw metadata (unrefined). The middle column shows the epipolar lines after the metadata were refined using the BA4S pipeline. The epipolar lines should ideally pass through the ground-truth points (center of the marked circles in each plot). As can be seen, the noise in metadata is significantly reduced after applying BA4S. The errors values in these plots are consistent with the EEE values in Fig. 42; see the pair (#50,#150) as well as the pair (#1,#158) in the matrix. The epipolar lines in the right column were

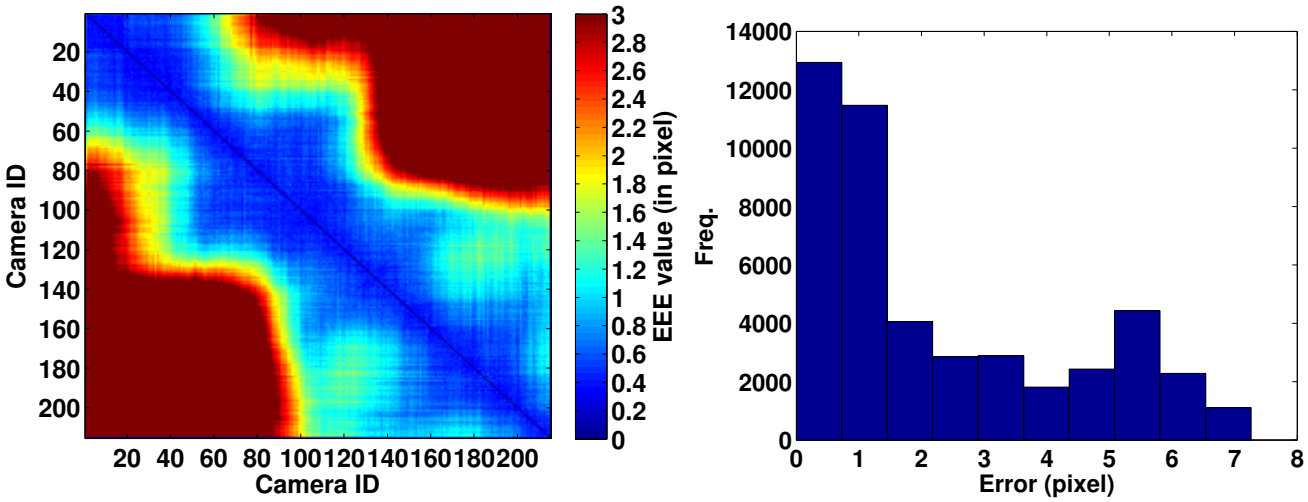
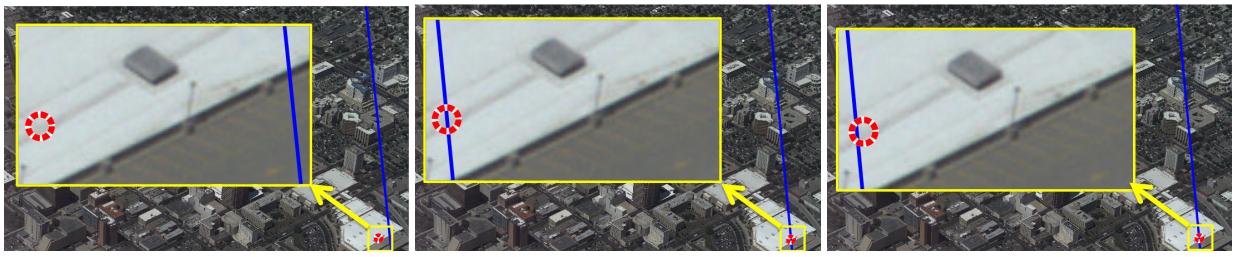
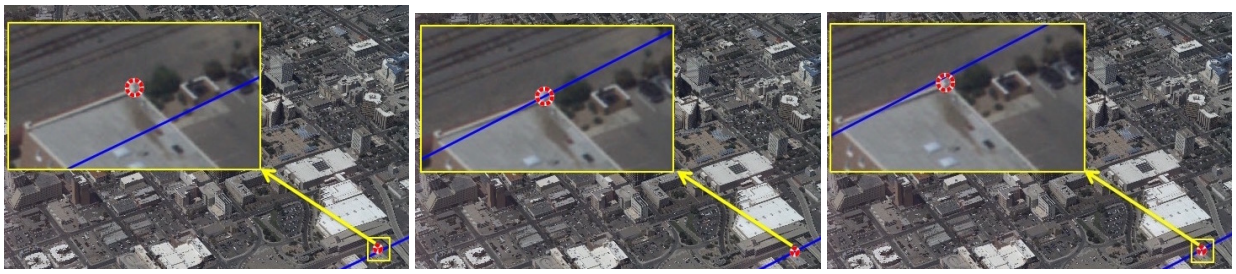


Figure 43: Evaluation of camera parameters using EEE measure after using MapTK for the ABQ215 aerial WAMI dataset. Left: EEE (error) plot. The pel in the matrix, ϵ_{ql}, indicates the error between the q-th camera (matrix rows) and l-th camera (columns), computed using (35). The ϵ_{ql} pel values were truncated to the maximum value of 3. The mean and standard deviation of the errors in pixels (36), over all cameras are $\mu_\epsilon = 2.31$ and $\sigma_\epsilon = 2$. Right: Histogram of the corresponding error values.



(a) ABQ215, frames (#50,#150). **Left**: uncorrected metadata. **Middle**: metadata corrected by BA4S. **Right**: VisualSfM



(b) ABQ215, frames (#1,#158). **Left**: uncorrected metadata. **Middle**: metadata corrected by BA4S. **Right**: VisualSfM

(c) Berkeley, frames (#50,#150). **Left**: uncorrected metadata. **Middle**: metadata corrected by BA4S. **Right**: VisualSfM

Figure 44: Visual assessment of camera parameters using epipolar lines using uncorrected metadata, refined metadata by BA4S and VisualSfM. Some ground-truth points between two pairs of cameras, (#50,#150) and (#1,#158), in the ABQ215 WAMI dataset and a pair of cameras (#50,#150) in Berkeley WAMI dataset were used. The shown images correspond to the left camera in each pair. On each shown image the corresponding ground-truth point is indicated by red circle. Epipolar lines corresponding to the ground-truth point from the right camera in each pair are directly calculated using camera parameters (using Eq. (28)) and plotted on the image of the left camera in each pair (shown images). The camera parameters from three different sources were used in each column; left: uncorrected original metadata, middle: BA4S (refined metadata) and right: VisualSFM.

(a) ABQ215 dataset. Left: uncorrected metadata with Euclidean error = 16.75 pixels. Right: Refined metadta by BA4S with Euclidean error = 0.23 pixels.

(b) Berkeley dataset. Left: uncorrected metadata with Euclidean error = 20.63 pixels. Right: Refined metadta by BA4S with Euclidean error = 0.14 pixels.

Figure 45: Visual assessment of camera parameters using epipolar lines using the metadata before and after refinement by BA4S. The left and right (cropped) image in each row correspond to the camera parameters assessments using original metadata (uncorrected) and BA4S optimized ones, respectively. While Fig. 44 depicted the epipolar lines for the point correspondences between just two views, each row in the current figure represents a similar assessment however between one camera and all other cameras within the dataset. The groundtruth point is marked in yellow. Instead of drawing all epipolar lines over an image, an analogue point for each epipolar line is plotted (colored in red). We obtain the line segment which joins the ground truth point to the epipolar line and is perpendicular to the line. Each of such a line segment is used as an error vector. The foot of the perpendicular corresponding to each epipolar is plotted as red points. The mean and covariance matrix of the error vectors are computed and the values (in pixels) are printed over the top of each image in the figure. The covariance matrix and mean are also plotted as an ellipse and point (in blue), respectively. In these two samples, we observed reductions in the errors by factors of more than 70 and 140 times respectively for Albuquerque and Berkeley datasets.

(a) VSFM- Euclidean Error = 2.02 pixel.

(b) BA4S- Euclidean Error = 0.82 pixel.

Figure 46: Visual assessment of camera parameters using epipolar lines using the corrected metadata by BA4S and VSFM. Visual assessment of camera parameters using epipolar ellipses in ABQ215 dataset (frame #215). Left: VSFM, right: BA4S. While Fig. 44 depicted the epipolar lines for the point correspondences between just two views, the current figure represents a similar assessment however between one camera and all other cameras within the dataset. The ground truth point is marked in yellow. Instead of drawing all epipolar lines over an image, an analogue point for each epipolar line is plotted (colored in red). We obtain the line segment which joins the ground truth point to the epipolar line and is perpendicular to the line. Each of such a line segment is used as an error vector. The foot of the perpendicular corresponding to each epipolar is plotted as red points. The mean and covariance matrix of the error vectors are computed and the values (in pixels) are printed over the top of each image in the figure. The covariance matrix and mean are also plotted as an ellipse and point (in blue), respectively. The Euclidean error corresponding to this sample is 2.02 and 0.82 pixels for VSFM and BA4S respectively. Notice that the red error points are all scattered very close to the ground truth (yellow point) and therefore may not be visible, specially for the BA4S result as the error is very small.

plotted using the camera parameters estimated by VisualSFM for comparison. The epipolar line for a ground truth point between a pair of cameras (#1,#158) is drawn. Similar plots are presented in Fig. 47 for comparison between the original metadata, and the corrected parameters by BA4S and MapTK. Another visual assessment is presented in Fig. 45. While Fig. 44 depicted the epipolar lines for the point correspondences between just two views, each row in Fig. 45 presents a similar assessment however between one camera and all other cameras in the dataset. The ground truth point is marked in yellow. Instead of drawing all epipolar lines over one image which would be very confusing, we have chosen to plot a representative point for each epipolar line (colored in red). We obtain the perpendicular line segment on each epipolar line which passes through the corresponding ground truth image point. The foot of such a perpendicular line on each epipolar line is used as a representative for the error between the two underlying cameras. Indeed, the length of such perpendicular line segment (from the foot to the ground truth point) was already used to compute ϵ_{ql} in (35). In figure 45, the foot of the perpendicular corresponding to each epipolar is plotted in red. In other words, the yellow point shows the ground truth point in the current frame (view/camera) and each red point indicates the error between the current frame (view/camera) and another frame (view/camera) in the sequence. The mean and covariance matrix of all errors are computed and the values (in pixels) are printed on the top of each image in the figure. The blue ellipse plotted on each image visualizes the corresponding covariance matrix, where the ellipse center is shown as a blue dot. The first and second rows of Fig. 45 correspond to an exemplary frame in ABQ215 and Berkeley datasets, respectively. Left images are for the cases where the original camera parameters (uncorrected metadata) were used and the right images are for the cases of using optimized camera parameters by BA4S. As one can see, the center of the covariance ellipse (blue point) for the BA4S ones nearly coincides with the ground truth point (yellow point) and the distribution of the errors is such trivial that the ellipse almost appears as a point, in contrary to the left images (original metadata ones). Similar plots can be seen in Fig. 46 for another frame (#214) for VSFM (left) and BA4S (right). The Euclidean error corresponding to this sample is 2.02 for VSFM and and 0.82 pixels for BA4S. Notice that the red error points are all scattered very close to the ground truth (yellow point) and therefore may not be visible, specially for the BA4S result as the error is very small.

Usually a dense 3D reconstruction algorithm such as PMVS[84] is applied after BA in order to obtain a dense and colored point cloud. We also applied PMVS for some of the datasets to visually assess reconstructed point clouds. The optimized metadata from BA4S is used as input to PMVS (or CMVS).

(a) Uncorrected metadata (b) Corrected by BA4S (c) Corrected by MapTK (less accurate)

Figure 47: Visual assessment of camera parameters using epipolar lines using uncorrected metadata, refined metadata by BA4S and MapTK. A ground-truth point between a pair of cameras (#1,#158) in the ABQ215 WAMI dataset is used. The shown images correspond to the left camera in each pair. On each shown image the corresponding ground-truth point is indicated by red circle. Epipolar lines corresponding to the ground-truth point from the right camera in each pair are directly calculated using camera parameters (using Eq. (28)) and plotted on the image of the left camera in each pair (shown images). The camera parameters from three different sources were used in each column; left: uncorrected original metadata, middle: BA4S (refined metadata) and right: MapTK [122]. As one can see there are some pixel errors between the ground truth point and corresponding epipolar line computed from MapTK result, which is consistent with the errors in plot 43-left in the region for the corresponding pair (between the pair #1,#158)

Figs 48-a, -b, -c and -d show the PMVS dense point clouds for ABQ215, Los Angeles, Berkeley and Columbia, respectively.

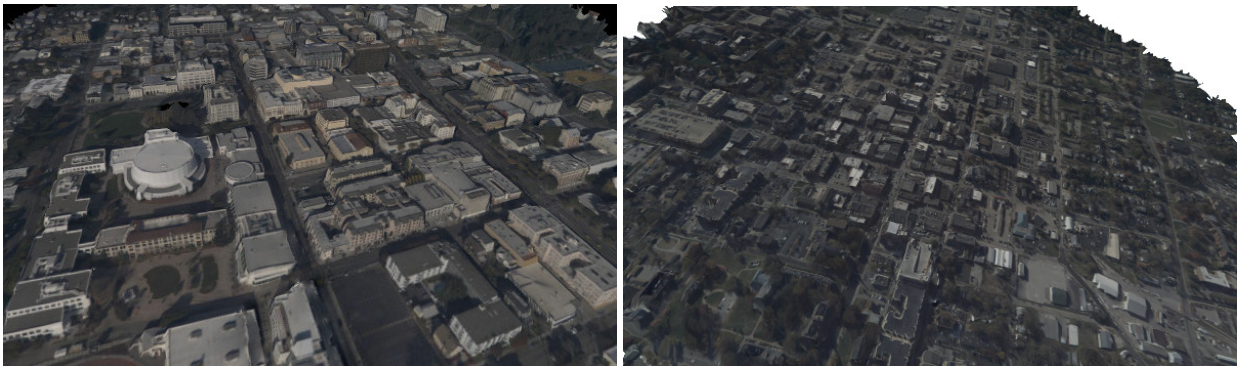
Non-WAMI datasets: In addition to testing BA4S on aerial WAMI datasets, we have applied it to the Middlebury benchmark datasets for multiview 3D reconstruction which are not WAMI but the images are acquired sequentially. *DinoRing* [186] is one of the challenging datasets for a classical BA due to the lack of salient image features for tracking across views [82, 13]. The proposed BA4S pipeline was tested on this dataset to evaluate its applicability and performance for non-WAMI trajectory images. The camera parameters in the Middlebury ground-truth were synthetically perturbed for both rotation and translation. The perturbed camera parameters along with the images were input to the BA4S pipeline. The *DinoRing* dataset has 48 cameras with image resolution of 640×480 pixels. The position errors for the metadata (perturbed camera parameters before optimization) and the optimized ones by BA4S are plotted in Fig. 49a. The first three rows in Fig. 50 show the visual assessment of three point correspondences. The errors for the corresponding epipolar lines are significantly reduced after BA4S optimized camera parameters. The epipolar lines have very large errors (the first and second columns of each row) when the noisy camera parameters are used. A dense version of the point cloud using PMVS with BA4S optimized camera parameters is shown in Fig. 51a. FountainP11 [199] is another non-WAMI dataset. As with the DinoRing dataset the ground-truth camera parameters were perturbed prior to running the BA4S algorithm. There are 11 cameras and each image has a resolution of 3072×2048 pixels. The position errors for the metadata (perturbed camera parameters) and the refined results using BA4S are plotted in Fig. 49b. The fourth, fifth and sixth rows in Fig. 50, each shows a point correspondences in this dataset. The initial epipolar lines have very large errors in the views (the first and second columns of each row) when the perturbed camera parameters were used. The errors for the corresponding epipolar lines become significantly smaller once BA4S refined camera parameters are used. A dense version of the point cloud for FountainP11 using PMVS and optimized camera parameters is shown in Fig. 51b.

4.2 Georegistration and stabilization

Refined camera parameters by BA4S were used to compute homography transformation matrices from each camera image plane onto the ground plane. Notice that such homographies were directly (analytically) computed using (25), as opposed to applying an image-to-image homography estimation method. Fig. 52 shows cropped areas of two registered images. Images of the first row correspond to a case where the original uncorrected metadata were used for registration. The second row shows the same frames, however registered by corrected metadata using the proposed pipeline. For each case, a few corresponding points (lying in the ground plane) between the two registered frames were manually picked and marked. As shown, the RMS error for the case where the uncorrected metadata were used is about 70

(a) Albuquerque, NM downtown

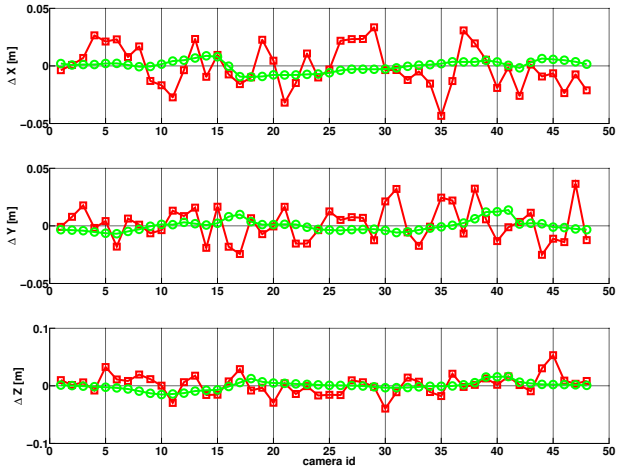
(b) Los Angeles, CA



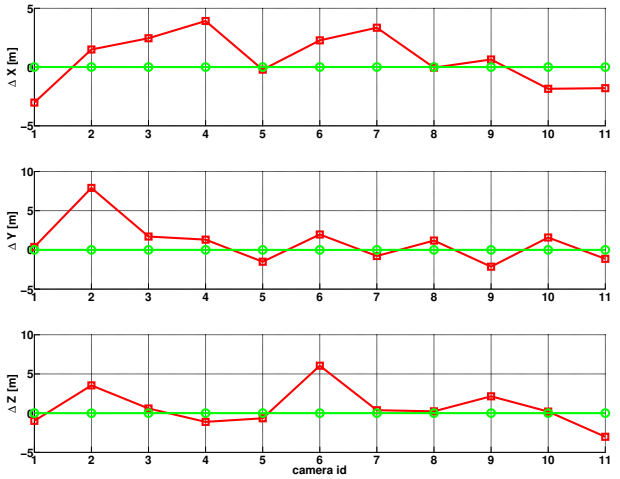
(c) Berkeley, CA

(d) Columbia, MO

Figure 48: Dense 3D point clouds for four aerial WAMI sequences using CMVS [81] with BA4S corrected camera pose metadata. Source imagery from Transparent Sky.



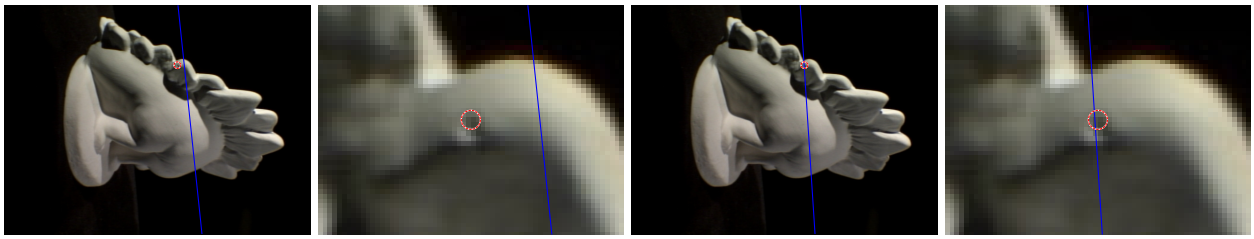
(a) DinoRing



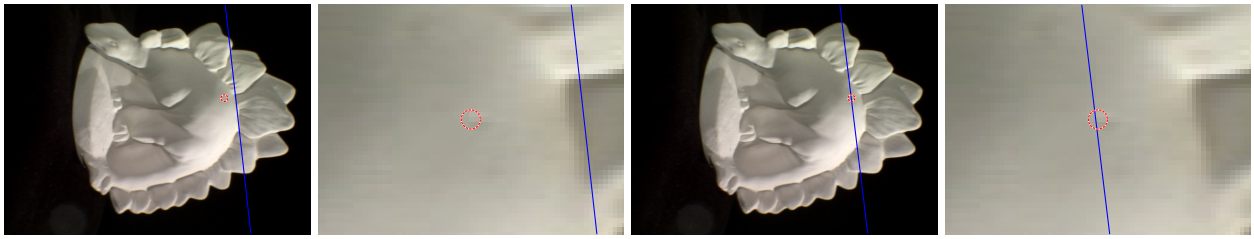
(b) FountainP11

Figure 49: Position errors before (red curve) and after (green curve) optimization using BA4S to refine the metadata for DinoRing (top) and Fountain-P11 (bottom) datasets.

(a) Metadata (b) Metadata-zoomed (c) After BA4S (d) BA4S-zoomed

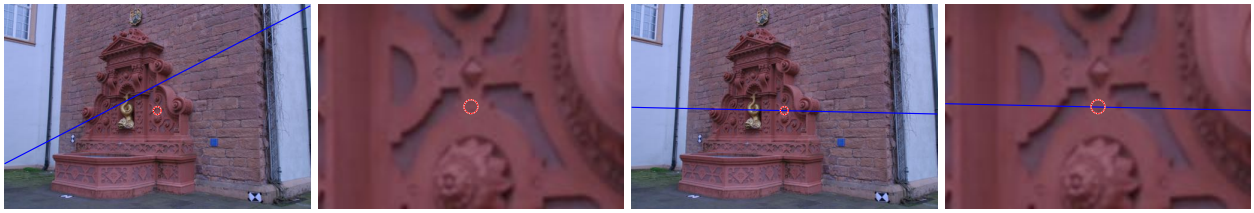


(e) Metadata (f) Metadata-zoomed (g) After BA4S (h) BA4S-zoomed



(i) Metadata (j) Metadata-zoomed (k) After BA4S (l) BA4S-zoomed

(m) Metadata (n) Metadata-zoomed (o) After BA4S (p) BA4S-zoomed



(q) Metadata (r) Metadata-zoomed (s) After BA4S (t) BA4S-zoomed

(u) Metadata (v) Metadata-zoomed (w) After BA4S (x) BA4S-zoomed

Figure 50: Visual assessment of camera parameters using epipolar lines in DinoRing (first three rows) and Fountain-P11 (last three rows) datasets. Each row shows the results for one correspondence. First and second columns are using the original (perturbed) camera metadata parameters. Notice that in (b), (f), (n), (r) and (v), the epipolar line went off the image plane due to large metadata error. Third and fourth columns of each row show the full and zoomed views using camera parameters after BA4S.

Figure 51: Dense reconstruction using PMVS after optimized camera parameters are estimated using BA4S. Left: DinoRing, right: FountainP11.

pixels. However, using the proposed pipeline this error was reduced to about 1.1 pixels.

Fig. 53 shows the overlays of the same registered frames. The overlays corresponding to the registration using the uncorrected metadata (plots on the left column) illustrate a mixed-up registration. However the overlays of registration using corrected metadata by BA4S (plots on the right column) demonstrate consistency between the results. As can be seen in the plots on the right column, only buildings were wobbled which is due to the parallax phenomenon explained in Sec 3.2. The points lying on the ground plane from two different views lined up perfectly which is the result of a precise registration by our proposed method. Unlike the image-to-image registration method in [130], in which the aerial images within each dataset could not be registered altogether (they were broken to several segments, see Table-I in [130]), our proposed method is able to efficiently register all the frames together with no fragmentations, thanks to the proposed robust optimization method.

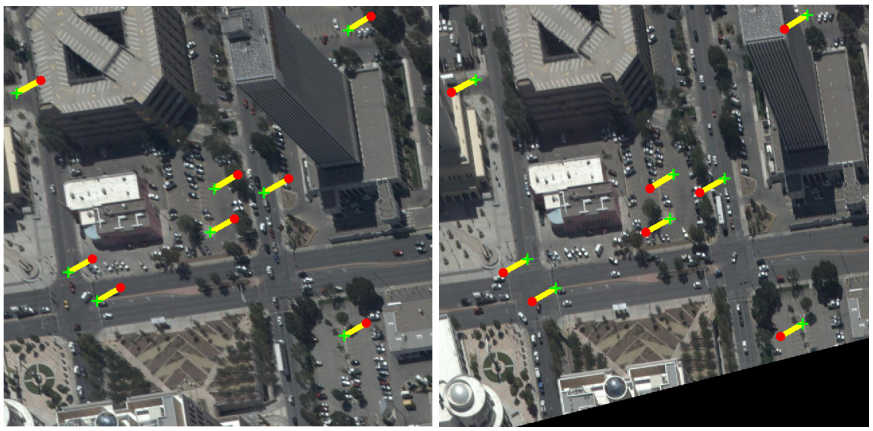
4.3 Vehicle Detection using Semantic Depth Map Fusion

In this Section we elaborate and evaluate our proposed vehicle moving object detection results for ABQ aerial urban imagery which were collected by TransparentSky [5] using an aircraft with on-board pose sensors flying over downtown Albuquerque, NM. Figure 21 shows samples of raw and geo-registered ultra high resolution images using BA4S state-of-the-art registration approach which processes them in very short amount of time (in this case less than 12 minutes for 1071 images). We worked on a cropped $2k \times 2k$ area of interest (AOI) for which the ground-truth are provided by Kitware (Fig. 54).

4.3.1 Moving Object Segmentation

The first input of our fusion scheme is the trace of the flux tensor matrix which provides information about temporal gradient changes or moving edges. Figure 54 shows the original cropped AOI and the flux trace results. every pixel is classified as moving versus stationary by thresholding trace of the corresponding flux tensor matrix. However, approximately 70% of the motion detection mask are in fact false detections which are technically posed by parallax motion affects of tall structures (Fig. 54).

We fused the altitude information of tall structure to the flux mask information in order to filter out the false detection caused by parallax motion effect of buildings. Figure 55 presents the fusion results of flux tensor motion detection and building mask. In the first raw, flux tensor motion detections are shown in 2 colors; false detections due to parallax are shown in red color and the rest are in yellow. In order to visually evaluate the results Ground truth bounding boxes are overlayed on flux mask in green color. Altitude mask corresponding to the ortho-rectified images are shown in second raw. all the pixels with altitude value greater than 20 are considered as tall structure and are shown as blue mask in the third raw. As discussed in Section 3.8.6 level-set based geodesic active contours is used to improve the building mask and reveal the filtered moving vehicles next to the buildings. Improved building mask contours and

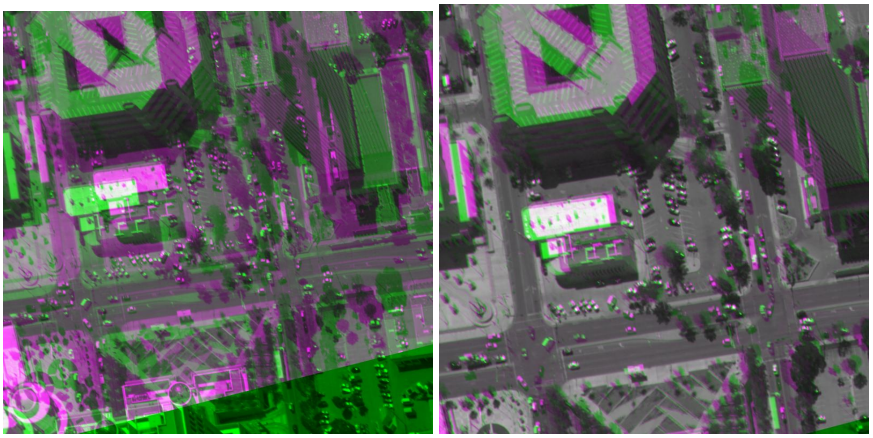


(a) Original metadata– Left: Frame #0. Right: Frame #60

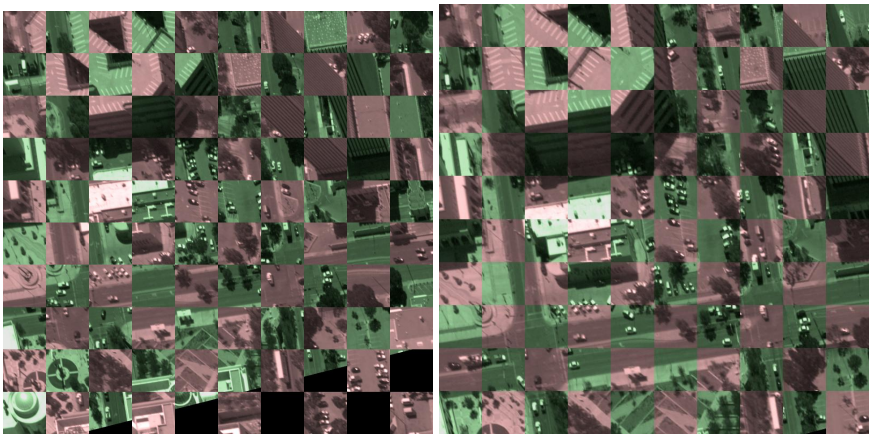
(b) Corrected by BA4S– Left: Frame #0. Right: Frame #60

Figure 52: Registration of frames #0 and #60 in ABQ WAMI dataset for two cases of uncorrected metadata and corrected by the proposed BA4S. For illustration, eight points within these cropped registered images were manually tracked. The green markers indicate the correct position of the points (ground truth) and the red ones indicate the corresponding points from the other frame after overlay. The average RMS error for the registered frames using the original metadata (uncorrected) and using corrected metadata (by BA4S) are about 70 and 1.1 pixels, respectively.

(a) Avg blending– Left: uncorrected metadata. Right: corrected by BA4S



(b) Purple-green blending– Left: uncorrected metadata. Right: corrected by BA4S



(c) Checkerboard overlay– Left: uncorrected metadata. Right: corrected by BA4S

Figure 53: Overlays of two registered frames (#0 and #60) in ABQ WAMI dataset for two cases of uncorrected metadata and corrected by the proposed BA4S. The corresponding registered frames were already shown in Fig. 52. As depicted on the right plots, the points lying on the ground plane from the two different frames are precisely coincided after BA4S registration. Pixels off the ground plane like the buildings are wobbled due to the parallax effects explained in Fig. 7.

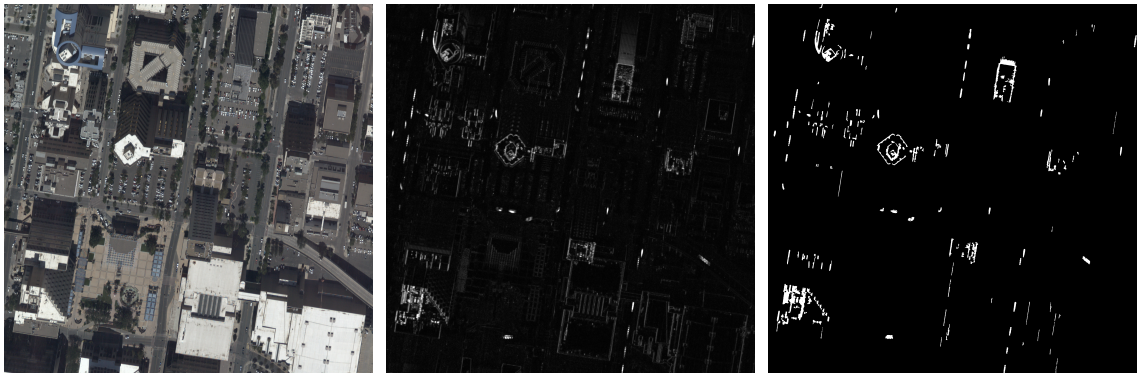


Figure 54: Illustration of motion detection using Flux trace only: From left to right, cropped ROI of Albuquerque aerial imagery (fr_{100}), the spatio-temporal motion information computed by flux trace for the selected image, flux mask in which each pixel is classified as moving or stationary by thresholding the flux trace. Morphology is applied to improve the result.

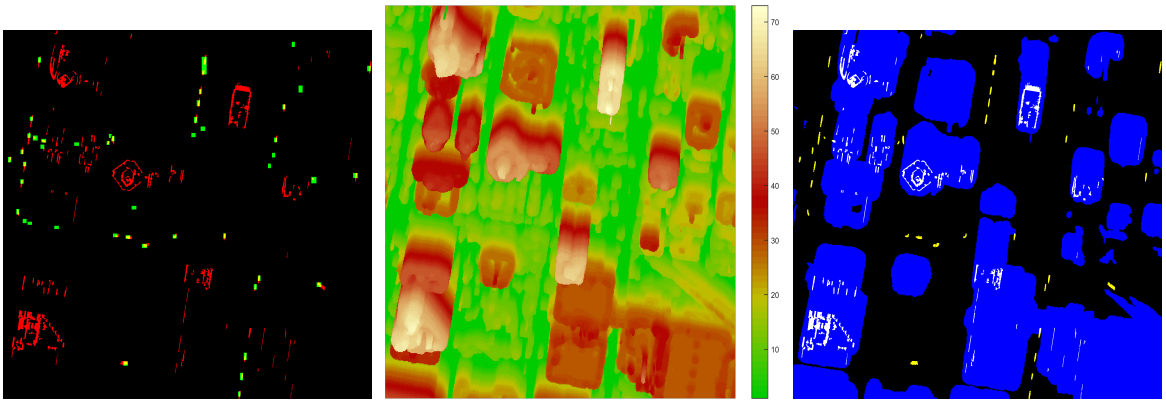


Figure 55: Illustration of motion detection using Flux trace and Depth

final motion detection results are shown in Figure 56.

4.3.2 Evaluation Methodology

Since the ultimate goal of the proposed motion detection system is to enable persistent tracking of moving vehicles, we have evaluated detection performance using object level measures. Associations of the detected moving blobs to ground truth objects is done using a bidirectional correspondence analysis described in [43, 89]) that handles not only one-to-one matches but also merge and fragmentation cases. Precision and Recall are computed at each stage of fusion as

$$Precision = \frac{N_{TrueDetection}}{N_{Detection}} = \frac{|TP|}{|TP| + |FP|} \tag{73}$$

$$Recall = \frac{N_{TrueDetection}}{N_{GT}} = \frac{|TP|}{|TP| + |FN|} \tag{74}$$

where $N_{TrueDetection}$ or *TP* is defined as total number of true one-to-one individual matches plus the number of Ground Truth fragmented objects and the number of merged detected objects. $N_{Detection}$ is the cardinality of detected objects and N_{GT} is total number of moving object bounding box presented in Ground Truth. We report $F_{measure}$ to evaluate the harmonic mean of recall and precision as well.

$$F - measure = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{75}$$

Figure 57 and 58 report the object level performance evaluation results.

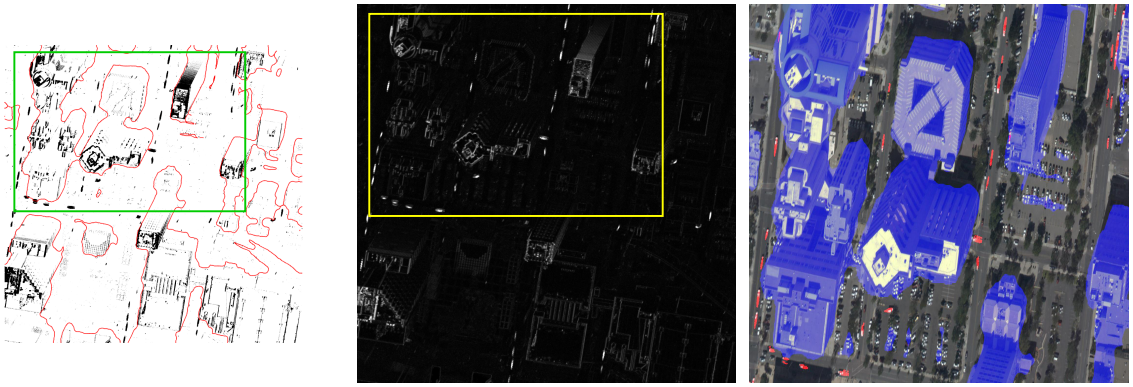


Figure 56: Illustration of motion detection using Flux trace+Depth+GAC

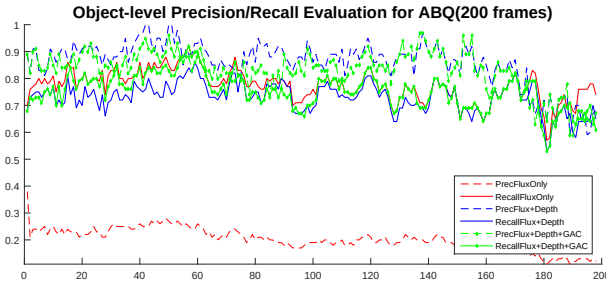


Figure 57: Object-level Precision and Recall: Performance evaluation of our proposed fused motion detection method

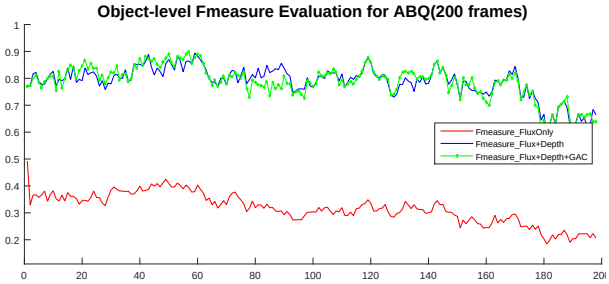
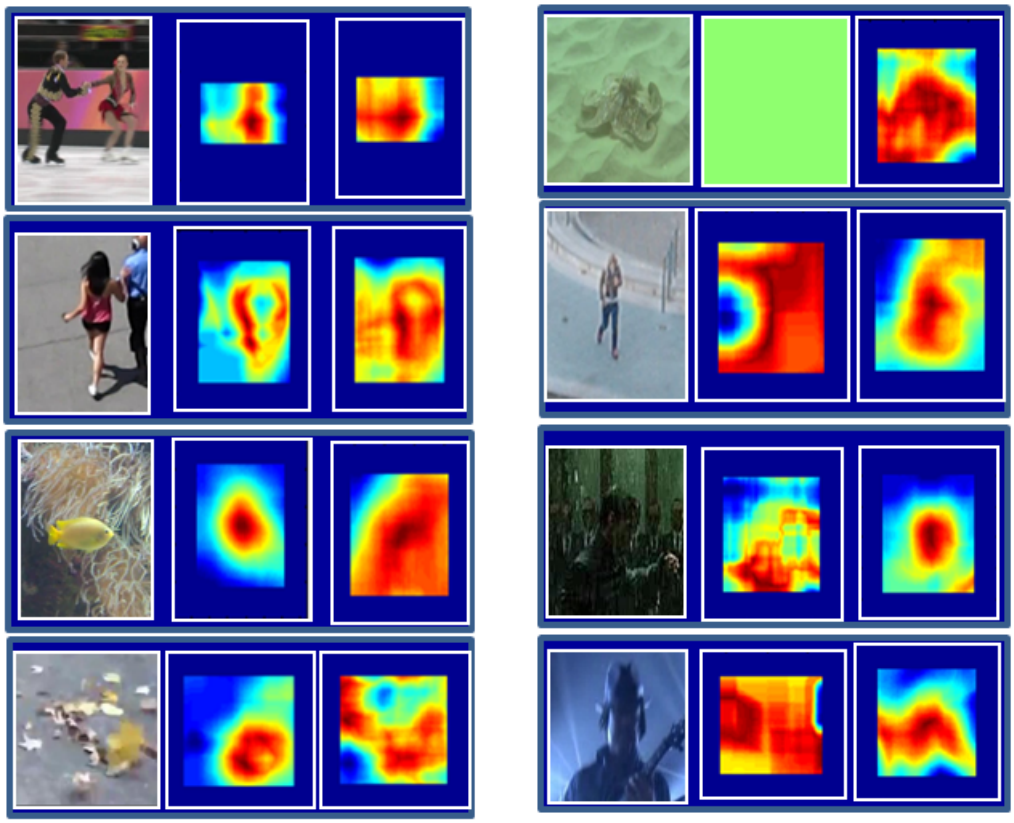


Figure 58: Object-Level $F_{measure}$: Performance evaluation of our proposed fused motion detection method

4.4 CS-LoFT Tracking

We have evaluated the CS-LoFT tracker on VOT2015 benchmark dataset [116], comparing against the original LoFT tracker. VOT challenge performance measures *accuracy* and *robustness* were used for evaluation. *Accuracy* measures how well the bounding box predicted by the tracker overlaps with the ground truth bounding box. *Robustness* measures how many times the tracker loses the target during tracking. A failure is indicated when the overlap measure becomes zero [116]. Figures 59 and 60 shows intermediate results related to color and scale processing in CS-LoFT. CS-LoFT switches to color tracking mode only when the tracked objects and their surroundings have different color features (Figure 59a), relies on intensity otherwise (Figure 59b). Figure 60 shows how much the proposed frame level max-pooling scale selection method preserves spatial context while pixel-wise max pooling over scales generates distractors. Table 3 and Figure 61 show detailed evaluation of LoFT and CS-LoFT on 60 video sequences of VOT2015 dataset. Table 3 groups the sequences into three sections according to their *robustness* measurement on CS-LoFT as improved, same, and worst. Figure 62 compares the performance of CN, our extended CN, LoFT, and the proposed CS-LoFT trackers in terms of robustness.

CS-LoFT improves the performance of the original LoFT tracker in terms of both *accuracy* and *robustness* by 11% and 23% respectively. Performance is particularly improved for the sequences that have



(a) (b)

Figure 59: Color versus intensity. (a) Tracked objects and their surroundings have different color features; (b) Tracked objects and their surroundings have similar color but different intensity features. Columns left to right: original image, likelihood map obtained using color names (CN) feature, likelihood map obtained using intensity feature.

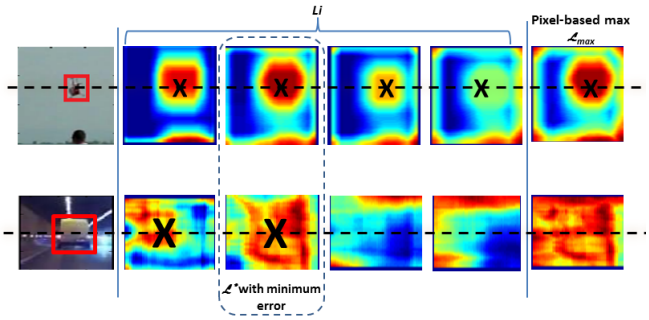
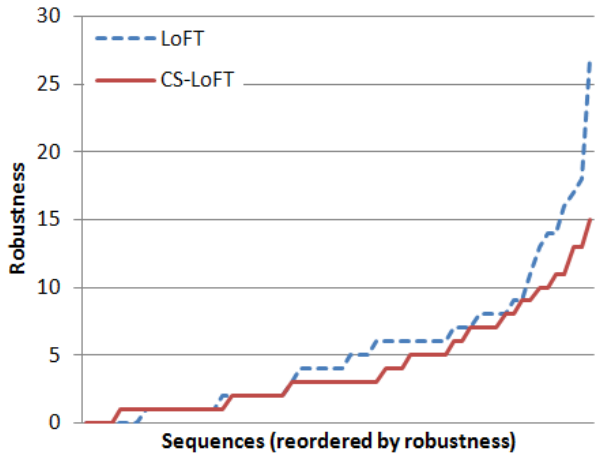


Figure 60: Frame level max-pooling best scale selection versus pixel-based max-pooling. The third column: the frame level $\mathcal{L}^*$ over all scales, while the sixth column: $\mathcal{L}_{max}$ pixel-based max-pooling result.

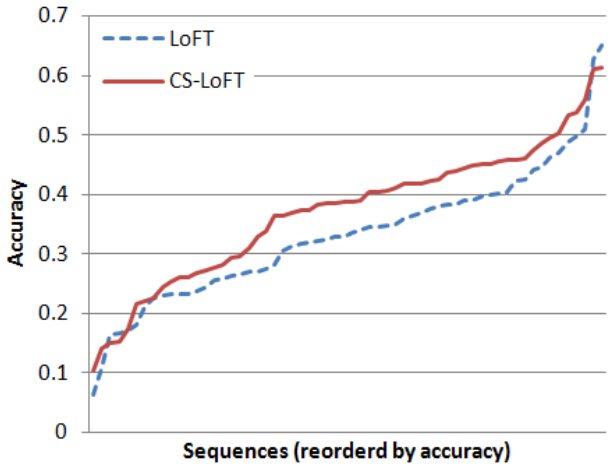
significant color distractor and rapid scale changes.

4.5 KC-LoFT Tracking

Benchmark datasets and challenges are important for fair evaluation and comparison of trackers. Since 2013, VOT challenge group [1] has been organizing single object tracking challenges for selected full motion video datasets. Table 4 shows VOT2015[116] and VOT2016[115] ranks of LoFT and some other state-of-the-art trackers. All the listed trackers perform better than LoFT on these full motion videos datasets. However, the nature of benchmark datasets can be one of the most important factors in tracker evaluation. LoFT and KC-LoFT are trackers developed for aerial wide area motion imagery. Aerial wide area motion imaginary (WAMI) datasets have different characteristics and challenges compared to



(a)



(b)

Figure 61: Performance evaluation on VOT2015 dataset: (a) robustness, (b) accuracy. Sequences reordered by robustness and accuracy values respectively.

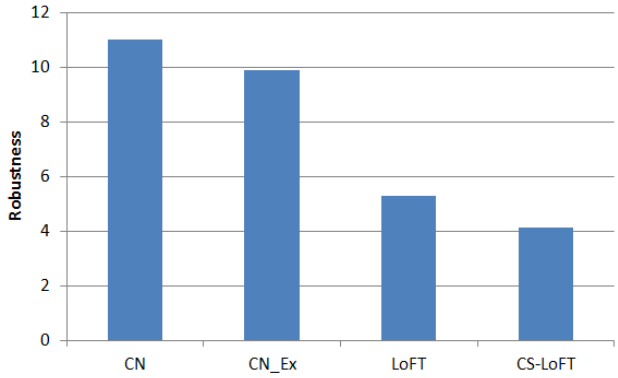


Figure 62: Robustness comparison between CN, extended CN, LoFT, and CS-LoFT trackers.

the regular full motion videos. These WAMI videos suffers from extreme camera motion, low frame rate, frequent object deformations, rapid scale and appearance changes, small object sizes, shadow and illumination artifacts, partial and full occlusions.

For this project, we have tested and evaluated the KC-LoFT tracker and the selected state-of-the-art trackers on the ABQ aerial wide area motion imagery dataset[2] for downtown Albuquerque, NM Figure 63. We have manually generated ground-truth tracks for 136 cars within a 200 frames subset of the ABQ dataset. A frame from the ABQ dataset and zoomed views of three sample cars from the scene are

Table 3: LoFT and CS-LoFT robustness and accuracy comparison ordered form most sequences that improved to less improved in robustness measurement.

No.	Sequences	LoFT	CS-LoFT	LoFT	CS-LoFT
		Robustness		Accuracy	
1	iceskater1	27	10	0.35	0.40
2	book	14	4	0.23	0.26
3	soccer1	17	7	0.45	0.49
4	iceskater2	14	5	0.35	0.41
5	bolt1	18	13	0.23	0.24
6	bolt2	6	3	0.23	0.25
7	hand	13	10	0.33	0.39
8	handball1	6	3	0.34	0.39
9	handball2	8	5	0.34	0.39
10	pedestrian1	6	3	0.38	0.44
11	ball1	6	4	0.11	0.14
12	dinosaur	4	2	0.26	0.28
13	fernando	4	2	0.26	0.29
14	fish4	5	3	0.27	0.34
15	gymnastics1	16	14	0.32	0.37
16	leaves	4	2	0.35	0.41
17	shaking	6	4	0.40	0.46
18	singer2	4	2	0.42	0.46
19	bag	1	0	0.06	0.10
20	fish2	6	5	0.27	0.31
21	gymnastics2	7	6	0.32	0.38
22	gymnastics4	6	5	0.33	0.38
23	motocross1	6	5	0.37	0.42
24	motocross2	4	3	0.37	0.42
25	rabbit	8	7	0.39	0.45
26	soldier	2	1	0.47	0.50
27	sphere	2	1	0.49	0.53
28	tunnel	2	1	0.63	0.61
29	ball2	2	2	0.16	0.15
30	birds2	1	1	0.18	0.22
31	blanket	2	2	0.21	0.22
32	crossing	0	0	0.26	0.28
33	fish1	4	4	0.26	0.30
34	godfather	1	1	0.31	0.37
35	graduate	11	11	0.32	0.37
36	gymnastics3	5	5	0.32	0.38
37	helicopter	1	1	0.34	0.40
38	nature	7	7	0.38	0.42
39	racing	0	0	0.40	0.45
40	road	7	7	0.40	0.45
41	sheep	2	2	0.40	0.46
42	singer3	1	1	0.44	0.47
43	soccer2	5	5	0.46	0.50
44	tiger	8	8	0.50	0.54
45	traffic	1	1	0.51	0.56
46	wiper	1	1	0.65	0.61
47	birds1	9	10	0.17	0.17
48	bmx	0	1	0.22	0.22
49	car1	0	1	0.24	0.27
50	car2	0	1	0.24	0.27
51	girl	8	9	0.28	0.36
52	glove	2	3	0.31	0.36
53	marching	2	3	0.36	0.42
54	octopus	0	1	0.38	0.44
55	pedestrian2	1	2	0.39	0.44
56	singer1	0	1	0.42	0.46
57	butterfly	3	5	0.23	0.26
58	fish3	1	3	0.27	0.33
59	matrix	9	11	0.36	0.42
60	basketball	6	9	0.17	0.15
Avg.		5.37	4.15	0.33	0.37

shown in figure 64. For tracker performance evaluation, we have used two VOT challenge measures ***Accuracy*** and ***Robustness***. ***Accuracy*** measures how well the bounding box predicted by a tracker overlaps with the ground truth bounding box. ***Robustness*** measures number of times a tracker looses the target during tracking. Table 5 summarizes the tracking performances for the proposed KC-LoFT tracker and other state-of-the-art trackers on ABQ dataset. KC-LoFT increases both accuracy and robustness of the original LoFT tracker (by 9.6% and 5.1% respectively) and produces better or comparable results compared to the state-of-the-art trackers from the VOT2015[116] and VOT2016[115] challenges. Figure 65 shows sample tracking results for two cars. Trajectory color represents number of reinitializations after tracker failures. Lower number of trajectory colors indicates tracker robustness. In both cases KC-LoFT tracks the selected cars without any failures or restarts, while for the first car LoFT, DSST, and Staple require one or more restarts, and for the second car all the trackers except KC-LoFT require one or more restarts.

4.6 Multi-object Tracking

We have tested and evaluated our MOT tracker on UA-DETRAC-test multi-object tracking benchmark dataset [211] consisting of 40 challenging real-world traffic videos. We have evaluated the performance of our tracker using the UA-DETRAC evaluation toolkit. The evaluation process includes the following met-

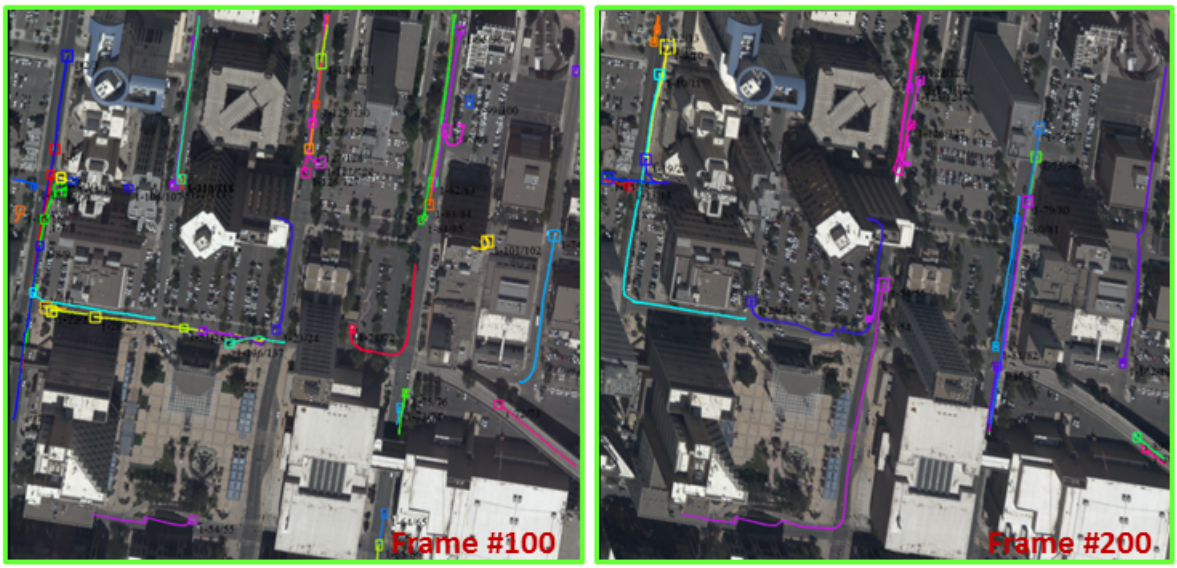


Figure 63: Sample KC-LoFT tracking results on ABQ aerial surveillance dataset.

Table 4: VOT2015 and VOT2016 ranks of the best non-deep learning trackers whose codes were made public. Lower rank is better. '-' means the tracker did not participate to the challenge.

Tracker Name	Rank on VOT2015	Rank on VOT2016
LoFT	#60	#68
STAPLE[34]	-	#5
SAMF [126]	#8	#24
DSST [60]	#38	#43
SRDCF [61]	#4	#17

Table 5: Tracker performance comparison on ABQ wide area motion imagery dataset. Bold: best result, underlined: second best result.

Tracker name	Accuracy↑	Robustness↓
SAMF[126]	**0.6396**	0.9197
DSST [60]	**0.6167**	1.7810
SRDCF[61]	0.4875	**0.7007**
STAPLE[34]	0.4972	1.1824
LoFT	0.5413	0.8540
KC-LoFT	0.5938	**0.8102**

rics described in [211]: mostly track (**PR-MT**), mostly lost (**PR-ML**), identity switches (**PR-IDS**), track fragmentation (**PR-FRAG**), false positives (**PR-FP**), false negatives (**PR-FN**), multi-object tracking precision (**PR-MOTP**), and multi-object tracking accuracy (**PR-MOTA**). Accuracy of the obtained tracks are best reflected by combination of the metrics **PR-IDS** and **PR-FRAG**. Final evaluation measures are computed by applying the trackers on a set of object detection results obtained by varying detection threshold, corresponding to different detection precision and recall levels.

We have combined our tracker on four different state-of-the-art object detectors, CompACT [46], R-CNN [85], ACF [67], and DPM [73]; and compared our tracking results to six state-of-the-art MOT trackers, including GOG [173], CEM [25], DCT [26], IHTLS [65], H^2T [212], and CMOT [29]. Table 6 summarizes the tracking performances of our tracker and other state-of-the-art trackers (described in [211]). Our tracker outperforms the other trackers in term of **PR-MOTA**, **PR-MOTP**, **PR-MT**, **PR-ML**, and **PR-FN**; produces comparable results for **PR-FRAG**, **PR-IDS**, and results in second highest score in terms of **PR-MOTA**, **PR-ML**, and **PR-FP**.

We have analyzed the contribution of different components to the overall performance of our tracker by systematically enabling/disabling different components (Figure 66). Our tests show that all components

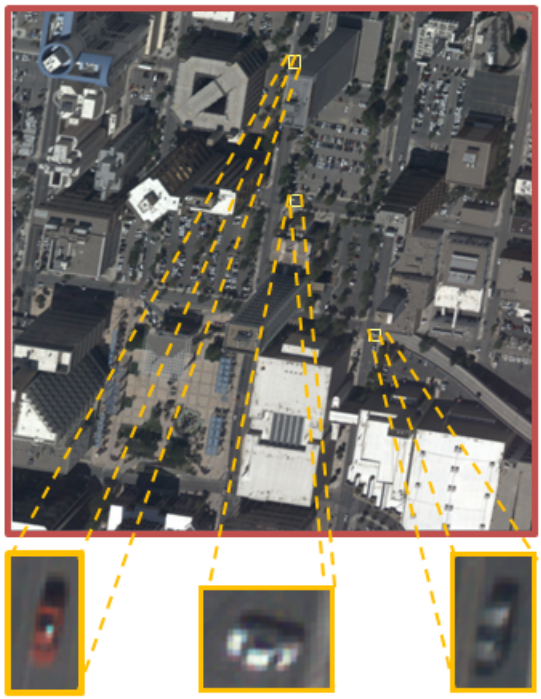


Figure 64: Sample car templates from an ABQ dataset frame.

Table 6: Tracker performance comparison using four state-of-the-art object detectors as input. Last raw indicates whether the metric for the specific column is better when high or low, + and − respectively. Best performance for each metric are marked in bold. We have two entries for our tracker based on how the performances scores are averaged. Challenge reports results in two groups corresponding to Easy versus (Medium + Hard) videos. Proposed-A computes average score as $(Score_{Easy} + Score_{Med+Hard})/2$. Proposed-B computes average score as $(0.25 \times Score_{Easy} + 0.75 \times Score_{Med+Hard})$ based on number of video sequences in each group. Scores are computed and averaged for all four object detectors according to [211].

Trackers	PR-MOTA	PR-MOTAP	PR-MT	PR-ML	PR-IDS	PR-FRAG	PR-FP	PR-FN
GOG [173]	10.1	35.3	10.9	22.5	4248.5	4137.3	43657.6	199926.8
CMOT [29]	7.0	34.6	12.8	**21.2**	414.3	1817.5	79577.3	187508.8
H^{2}T [212]	7.8	34.6	11.2	22.2	1298.9	1477.4	65275.3	196107.0
DCT [26]	8.3	35.6	5.5	32.3	**270.1**	**261.4**	**17658.2**	241590.8
IHTLS [65]	5.8	35.1	9.6	23.4	1329.2	4597.1	68635.0	205649.3
CEM [25]	3.9	33.6	2.4	36.1	394.3	529.0	19044.8	267699
SCTrack(Ours)-A	**12.8**	**37.7**	**12.9**	21.9	494.9	1408.0	27581.0	**99642.3**
SCTrack(Ours)-B	9.8	35.3	10.96875	22.637	656.48	1408.08	35945.59	130496.6
Better	+	+	+	-	-	-	-	-

contribute to better accumulated performance. For example, in $Experiment - E$, disabling the CN-to-CN color correlation weight matrix $\mathcal{W}_{CN}$ and assuming uncorrelated color codes, results in a drop in PR-MOTA metric from 13.1 to 12.7. Figure 67 shows performance improvement in the tracker by addition of each component.

Table 7 summarizes frame rates (in terms of frames per seconds) for our tracker and other state-of-the-art trackers provided in [211]. Frame rate for our tracker has been obtained averaging frame rates on 40 sequences of UA-DETRAC-test set. During the tests, CompACT object detection results have been used as input. Our simple feature set combined with two-step distance-only local and distance appearance combined global data association scheme allows high frame rates while still preserving reliability and accuracy of our tracker.

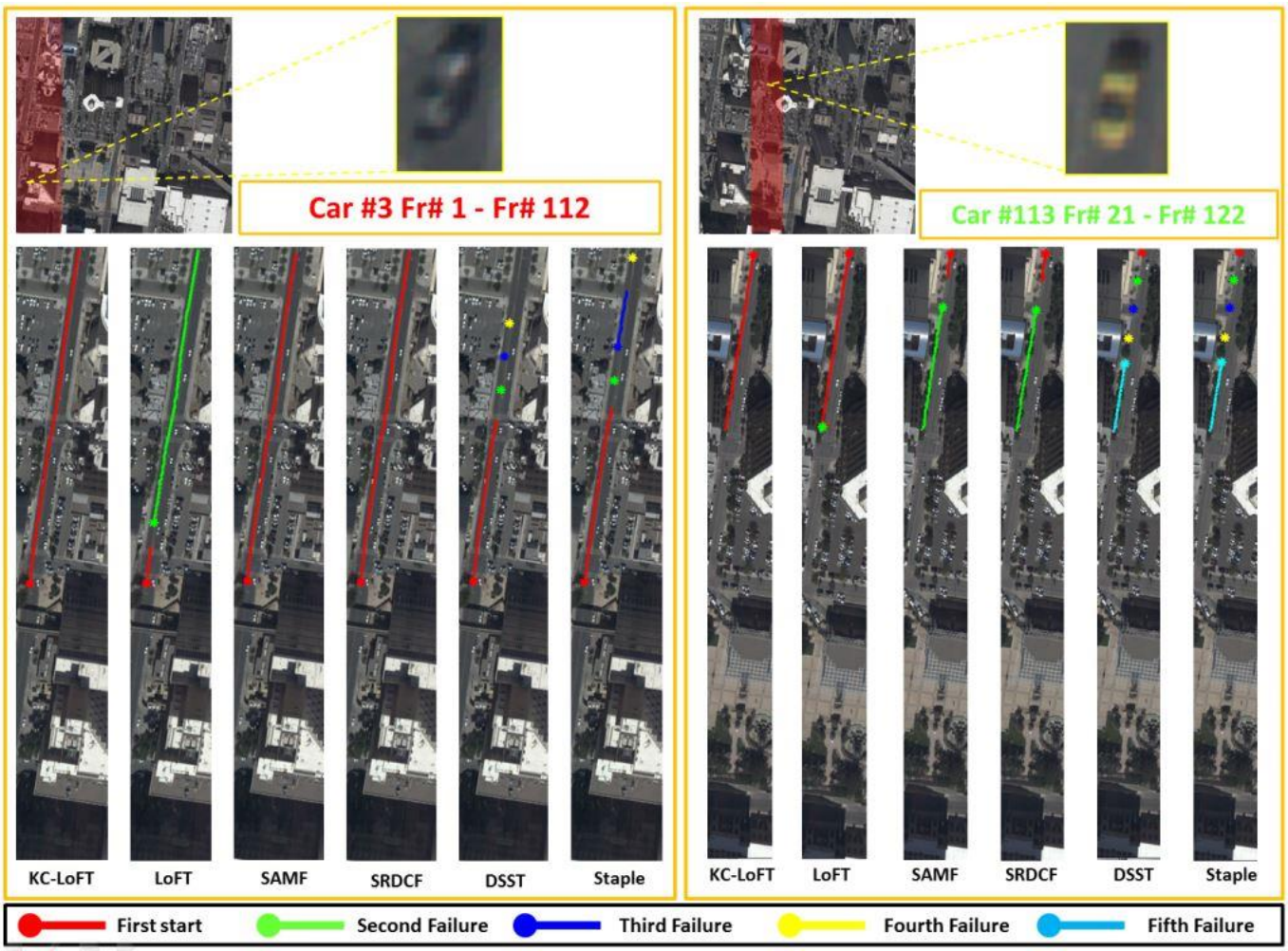


Figure 65: Comparison of tracking results in terms of robustness. Trajectory color represents number of reinitializations after tracker failures. Lower number of trajectory colors indicates higher tracker robustness.

5 Conclusions

In this effort we have worked on accomplishing the goals of the project which was to push the computation closer to the sensor. In this direction we developed novel and effective methods for feature detection and matching techniques along the video sequence, structure-from-motion and bundle adjustment for high resolution aerial imagery, 3D reconstruction algorithm to run on metropolitan-scale data, analytical image stabilization and georegistration, and moving object detection and tracking algorithms.

Table 7: Frame rate comparison.

Trackers	Code	Frame Rate (fps)
CEM	Matlab	4.62
GOG	Matlab	**389.51**
DCT	Matlab, C++	2.19
IHTL	Matlab	19.79
H2T	C++	3.02
CMOT	Matlab	3.79
SCTrack(Ours)	Matlab	362.00

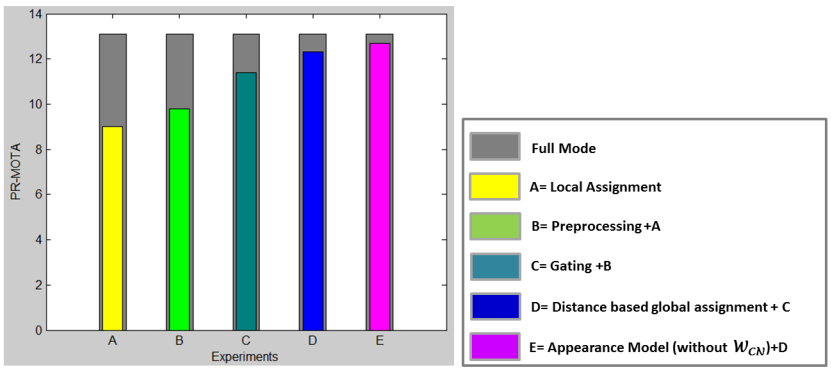


Figure 66: Contribution of each tracker component. Left: PR-MOTA metric for each experiment compared to the Full Mode (higher is better). Right: description of each experiment.

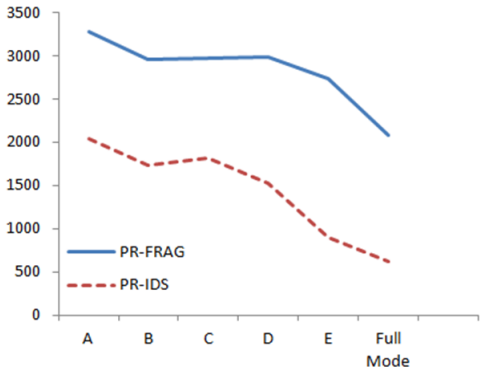


Figure 67: Contribution of each tracker component on PR-IDS and PR-FRAG metrics (lower is better).

6 LIST OF SYMBOLS ABBREVIATIONS AND ACRONYMS

- Wide Area Motion Imagery (WAMI)
- Bundle Adjustment (BA)
- Bundle Adjustment for Sequential Imagery (BA4S)
- Structure from Motion (SfM)
- Likelihood of Features Tracking (LOFT)
- Kernelized Correlation Extended LoFT for Single Object Tracking (KC-LoFT)
- ICCV Video Object Tracking Challenge (VOTC)
- Ground sampling distance (GSD)
- Levenberg-Marquardt (LM)
- Unmanned Aerial Vehicle (UAV)
- Epipolar Transformation Visualization Tool (EpiX)
- Simultaneous Localization And Mapping (SLAM)
- Kalman Filter (KF)
- Extended Kalman Filter (EKF)
- Unscented Kalman Filter (UKF)
- Sequential Monte-Carlo (SMC)
- Maximum Likelihood (ML)
- Expectation Maximization (EM)
- Bilateral Filter (BLF)

References

[1] http://www.votchallenge.net. 79

[2] http://www.transparentsky.net. 80

[3] Agisoft, Agisoft Photoscan Professional. http://www.agisoft.com. 20

[4] https://www.google.com/atap/project-tango. 13

[5] http://www.transparentsky.net. 6, 39, 60, 63, 74

[6] N. Snavely, Bundler: Structure from Motion (SfM) for Unordered Image Collections. http://phototour.cs.washington.edu/bundler. 20

[7] Pix4D, http://pix4d.com. 10, 20, 60, 65

[8] S. Agarwal, Y. Furukawa, and N. Snavely. Building Rome in a day. *Communications of the ACM*, 54:105–112, 2011. 13, 15

[9] S. Agarwal and K. Mierle. Ceres solver. http://ceres-solver.org. 60

[10] S. Agarwal, N. Snavely, S. M. Seitz, and R. Szeliski. Bundle adjustment in the large. In *European Conference Computer Vision*, pages 29–42, 2013. 15

[11] N. M. Al-Shakarji, F. Bunyak, G. Seetharaman, and K. Palaniappan. Robust multi-object tracking with semantic color correlation. In *IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 1–7, 2017. 44

[12] N. M. AL-Shakarji, F. Bunyak, G. Seetharamany, and K. Palaniappan. Cs-loft: Color and scale adaptive tracking using max-pooling with bhattacharyya distance. In *Applied Imagery Pattern Recognition Workshop*, pages 1–7, 2016. 49

[13] A. Albarelli, E. Rodola, and A. Torsello. Imposing semi-local geometric constraints for accurate correspondences selection in structure from motion: A game-theoretic perspective. *Int. Journal of Computer Vision*, 97:36–53, 2012. 18, 60, 70

[14] H. Aliakbarpour, L. Almeida, P. Menezes, and J. Dias. Multi-sensor 3D volumetric reconstruction using CUDA. *3D Research, Springer*, 2(4):1–14, 2011. 13

[15] H. Aliakbarpour and J. Dias. IMU Aided 3D Reconstruction Based on Multiple Virtual Planes. In *2010 International Conference on Digital Image Computing: Techniques and Applications*, pages 474–479. IEEE, dec 2010. 13

[16] H. Aliakbarpour and J. Dias. PhD forum: Volumetric 3D reconstruction without planar ground assumption. In *2011 Fifth ACM/IEEE International Conference on Distributed Smart Cameras*, pages 1–2. IEEE, aug 2011. 13

[17] H. Aliakbarpour and J. Dias. Three-dimensional reconstruction based on multiple virtual planes by using fusion-based camera network. *Jour. of IET Computer Vision*, 6(4):355, 2012. 13

[18] H. Aliakbarpour, K. Palaniappan, and J. Dias. Geometric exploration of virtual planes in a fusion-based 3D data registration framework. In *Proc. SPIE Conf. Geospatial InfoFusion III (Defense, Security and Sensing: Sensor Data and Information Exploitation)*, volume 8747, page 87470C, 2013. 13

[19] H. Aliakbarpour, K. Palaniappan, and J. Dias. Geometric exploration of virtual planes in a fusion-based 3d registration framework. In *Proc. SPIE Conf. Geospatial InfoFusion III (Defense, Security and Sensing: Sensor Data and Information Exploitation)*, volume 8747, page 87470C, 2013. 39

[20] H. AliAkbarpour, K. Palaniappan, and G. Seetharaman. Fast structure from motion for sequential and wide area motion imagery. *Proc. IEEE International Conference on Computer Vision Workshop (ICCVW) Video Summarization for Large-scale Analytics Workshop*, Dec 2015. 38, 39

[21] H. Aliakbarpour, K. Palaniappan, and G. Seetharaman. Robust camera pose refinement and rapid SfM for multi-view aerial imagery without RANSAC. *IEEE Journal of Geoscience and Remote Sensing Letters*, 12(11):2203–2207, 2015. 19

[22] H. Aliakbarpour, K. Palaniappan, and G. Seetharaman. Robust camera pose refinement and rapid SfM for multiview aerial imageryâĂŤwithout ransac. *IEEE Geoscience and Remote Sensing Letters*, 12(11):2203–2207, Nov 2015. 38, 39

[23] H. Aliakbarpour, V. B. S. Prasath, K. Palaniappan, G. Seetharaman, and J. Dias. Heterogeneous Multi-View Information Fusion: Review of 3-D Reconstruction Methods and a New Registration with Uncertainty Modeling. *IEEE Access*, 4:8264–8285, 2016. 13

[24] L. Alvarez, R. Deriche, J. Sanchez, and J. Weickert. Dense disparity map estimation respecting image discontinuities: A {PDE} and scale-space based approach. *Journal of Visual Communication and Image Representation*, 13(1-2):3 – 21, 2002. 57

[25] A. Andriyenko and K. Schindler. Multi-target tracking by continuous energy minimization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1265–1272, 2011. 82, 83

[26] A. Andriyenko, K. Schindler, and S. Roth. Discrete-continuous optimization for multi-target tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1926–1933, 2012. 82, 83

[27] A. Aravkin, M. Styer, Z. Moratto, A. Nefian, and M. Broxton. Student's t robust bundle adjustment algorithm. In *International Conference on Image Processing*, pages 1757–1760, 2012. 18

[28] G. Aubert and P. Kornprobst. *Mathematical problems in image processing: Partial differential equation and calculus of variations*. Springer-Verlag, New York, USA, 2006. 54

[29] S.-H. Bae and K.-J. Yoon. Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1218–1225, 2014. 82, 83

[30] Y. Ban, S. Ba, X. Alameda-Pineda, and R. Horaud. Tracking multiple persons based on a variational bayesian model. In *European Conference on Computer Vision*, pages 52–67, 2016. 44

[31] A. Basharat, M. Turek, Y. Xu, C. Atkins, D. Stoup, K. Fieldhouse, P. Tunison, and A. Hoogs. Real-time multi-target tracking at 210 megapixels/second in wide area motion imagery. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 839–846, 2014. 38

[32] T. Benlefki, R. Liu, B. Dai, and M. B. Boubekeur. Combining color and vibe foreground mask for better tracking in video sequences. *Int. Cong. on Image and Signal Processing*, pages 18–22, 2015. 46

[33] B. Berlin and P. Kay. Basic color terms: their universality and evolution. *University of California Press*, 1969. 46

[34] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. H. Torr. Staple: Complementary learners for real-time tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1401–1409, 2016. 49, 82

[35] E. W. Bethel. Exploration of Optimization Options for Increasing Performance of a GPU Implementation of a Three-Dimensional Bilateral Filter. Technical Report LBNL-5406E, Lawrence Berkeley National Laboratory, Berkeley, CA, USA, 94720, 2012. 54

[36] S. Bhattacharya, H. Idrees, I. Saleemi, S. Ali, and M. Shah. Moving object detection and tracking in forward looking infra-red aerial imagery. In *Machine Vision Beyond Visible Spectrum*, pages 221–252. Springer, 2011. 38

[37] A. Bhattacharyya. On a measure of divergence between two multinomial populations. *SankhyÄĄ: The Indian Journal of Statistics (1933-1960)*, 7(4):401–406, 1946. 46

[38] E. Blasch, P. Deignan, S. Dockstader, M. Pellechia, K. Palaniappan, and G. Seetharaman. Contemporary concerns in geographical/geospatial information systems (GIS) processing. In *Proc. IEEE National Aerospace and Electronics Conference (NAECON)*, pages 183–190, 2011. 13

[39] E. Blasch, G. Seetharaman, K. Palaniappan, H. Ling, and G. Chen. Wide-area motion imagery (WAMI) exploitation tools for enhanced situational awareness. In *Proc. IEEE Applied Imagery Pattern Recognition Workshop*, 2012. 13

[40] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui. Visual object tracking using adaptive correlation filters. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2544–2550, 2010. 44, 46, 49, 51

[41] M. Bryson, A. Reid, F. Ramos, and S. Sukkarieh. Airborne vision-based mapping and classification of large farmland environments. *Journal of Field Robotics*, 27(5):632–655, may 2010. 15

[42] A. Buades, B. Coll, and J. M. Morel. A review of image denoising methods, with a new one. *Multiscale Modeling and Simulation*, 4(2):490–530, 2006. 54

[43] F. Bunyak, A. Hafiane, and K. Palaniappan. Histopathology tissue segmentation by combining fuzzy clustering with multiphase vector level sets. In *Software tools and algorithms for biological systems*, pages 413–424. Springer, 2011. 77

[44] F. Bunyak, K. Palaniappan, S. K. Nath, T. Baskin, and G. Dong. Quantitative cell motility for in vitro wound healing using level set-based active contour tracking. In *IEEE International Symposium on Biomedical Imaging: Nano to Macro*, pages 1040–1043, 2006. 44

[45] F. Bunyak, K. Palaniappan, S. K. Nath, and G. Seetharaman. Flux tensor constrained geodesic active contours with sensor fusion for persistent object tracking. *J. Multimedia*, 2(4):20–33, Aug 2007. 40

[46] Z. Cai, M. Saberian, and N. Vasconcelos. Learning complexity-aware cascades for deep pedestrian detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3361–3369, 2015. 82

[47] F. Calakli and G. Taubin. SSD: Smooth signed distance surface reconstruction. *Computer Graphics Forum*, 30(7):1993–2002, 2011. 55

[48] V. Carletti, P. Foggia, A. Greco, A. Saggese, and M. Vento. Automatic detection of long term parked cars. In *12th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6, 2015. 38

[49] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic active contours. *International journal of computer vision*, 22(1):61–79, 1997. 43

[50] M. Centir, P. Fragneto, D. Denaro, B. Rossi, and C. Marchisio. A combined color-correlation visual model for object tracking using particle filters. *Int. Symp. on Image and Signal Processing and Analysis*, pages 153–158, 2013. 46

[51] B. Chakraborty, M. B. Holte, T. B. Moeslund, and J. Gonzàlez. Selective spatio-temporal interest points. *Computer Vision and Image Understanding*, 116(3):396–410, 2012. 38

[52] T. F. Chan and L. A. Vese. Active contours without edges. *IEEE transactions on Image processing*, 10(2):266–277, 2001. 43

[53] S. Chaves, R. Wolcott, and R. Eustice. NEEC ResearchâĂŕ: Toward GPS-denied landing of unmanned aerial vehicles on ships at sea. *Naval Engineers Journal*, pages 105–112, 2014 (In Press). 59

[54] V. M. Chellappa, Govindu, and R. Feature-based image to image registration. In *Image Registration for Remote Sensing*, pages 215–239. 2011. 20

[55] B.-J. Chen and G. Medioni. Motion propagation detection association for multi-target tracking in wide area aerial surveillance. In *IEEE Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6, 2015. 38

[56] R. T. Collins, Y. Liu, and M. Leordeanu. Online selection of discriminative tracking features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1631–1643, 2005. 43, 49

[57] D. Crispell, J. Mundy, and G. Taubin. Parallax-free registration of aerial video. In *Proceedings of the British Machine Vision Conference*, pages 73.1–73.10. BMVA Press, 2008. doi:10.5244/C.22.73. 39

[58] D. Crispell, J. L. Mundy, and G. Taubin. Parallax-free registration of aerial video. *British Machine Vision Conference*, pages 1–4, 2008. 20, 21

[59] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 886–893, 2005. 51, 54

[60] M. Danelljan, G. Häger, F. Khan, and M. Felsberg. Accurate scale estimation for robust visual tracking. In *British Machine Vision Conference*, 2014. 46, 49, 82

[61] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg. Learning spatially regularized correlation filters for visual tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4310–4318, 2015. 82

[62] M. Danelljan, F. Shahbaz Khan, M. Felsberg, and J. Van de Weijer. Adaptive color attributes for real-time visual tracking. *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1090–1097, 2014. 46

[63] A. Delaunoy and M. Pollefeys. Photometric bundle adjustment for dense multi-view 3D modeling. In *IEEE Conf. Computer Vision Pattern Recognition*, pages 1486–1493, 2014. 15

[64] T. Dickscheid, T. Läbe, and W. Förstner. Benchmarking automatic bundle adjustment results. In *21st Congress of the International Society for Photogrammetry and Remote Sensing (ISPRS). Beijing, China*, pages 7–12, 2008. 15

[65] C. Dicle, O. I. Camps, and M. Sznaier. The way they move: Tracking multiple targets with similar appearance. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2304–2311, 2013. 82, 83

[66] T. B. Dinh, N. Vo, and G. Medioni. Context tracker: Exploring supporters and distracters in unconstrained environments. *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1177–1184, 2011. 46

[67] P. Dollár, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 36(8):1532–1545, 2014. 82

[68] A. G. (ed) and T. S. Huang. Calibration and Orientation of Cameras in Computer Vision, 2002. 18

[69] S. Y. Elhabian, K. M. El-Sayed, and S. H. Ahmed. Moving object detection in spatial domain using background removal techniques-state-of-art. *Recent patents on computer science*, 1(1):32–54, 2008. 38

[70] I. Ersoy, K. Palaniappan, and G. Seetharaman. Visual tracking with robust target localization. In *IEEE Int. Conf. Image Processing*, pages 1365–1368, 2012. 38

[71] I. Ersoy, K. Palaniappan, G. Seetharaman, and R. Rao. Interactive tracking for persistent wide-area surveillance. In *Proc. SPIE Conf. Geospatial InfoFusion II (Defense, Security and Sensing: Sensor Data and Information Exploitation)*, volume 8396, 2012. 38

[72] M. E. Farmer, X. Lu, H. Chen, and A. K. Jain. Robust motion-based image segmentation using fusion. In *IEEE International Conference on Image Processing*, volume 5, pages 3375–3378, 2004. 38

[73] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Ttransactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010. 82

[74] R. Feris, R. Bobbitt, S. Pankanti, and M.-T. Sun. Efficient 24/7 object detection in surveillance videos. In *IEEE Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6, 2015. 38

[75] S. Fleishman, I. Drori, and D. Cohen-Or. Bilateral mesh denoising. *ACM Trans. Graph.*, 22(3):950–953, 2003. 54, 55

[76] P. Foggia, A. Saggese, N. Strisciuglio, M. Vento, and N. Petkov. Car crashes detection by audio analysis in crowded roads. In *12th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6, 2015. 38

[77] K. Fragkiadaki, P. A. Arbeláez, P. Felsen, and J. Malik. Spatio-temporal moving object proposals. *arXiv preprint arXiv:1412.6504*, 2014. 38

[78] J.-M. Frahm, M. Pollefeys, S. Lazebnik, D. Gallup, B. Clipp, R. Raguram, C. Wu, C. Zach, and T. Johnson. Fast robust large-scale mapping from video and internet photo collections. *ISPRS Journal of Photogrammetry and Remote Sensing*, 65(6):538–549, 2010. 15

[79] F. Fraundorfer and D. Scaramuzza. Visual odometry: Part ii: Matching, robustness, optimization, and applications. *IEEE Robotics & Automation Magazine*, 19(2):78–90, 2012. 15, 18, 64

[80] G. Fumera and F. Roli. A theoretical and experimental analysis of linear combiners for multiple classifier systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(6):942–956, 2005. 49

[81] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. Towards Internet-scale multi-view stereo. In *IEEE Computer Vision and Pattern Recognition Conference*, pages 1434–1441, 2010. 4, 8, 14, 71

[82] Y. Furukawa and J. Ponce. Accurate camera calibration from multi-view stereo and bundle adjustment. *Int. Journal of Computer Vision*, 84(3):257–268, 2009. 23, 60, 70

[83] Y. Furukawa and J. Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8):1362–1376, 2010. 39

[84] Y. Furukawa and J. Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 32(8):1362–76, 2010. 69

[85] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014. 82

[86] A. González, R. Martín-Nieto, J. Bescós, and J. M. Martínez. Single object long-term tracker for smart control of a ptz camera. In *Proceedings of the International Conference on Distributed Smart Cameras*, page 39, 2014. 44

[87] G. Graber, T. Pock, and H. Bischof. Online 3D reconstruction using convex optimization. In *IEEE Int. Conf. Computer Vision Workshops (ICCV Workshops)*, pages 708–711, 2011. 55, 57

[88] M. Grabner. Object recognition based on local feature trajectories. *Proceedings of the International Cognitive Vision*, 2005. 16

[89] A. Hafiane, F. Bunyak, and K. Palaniappan. Fuzzy clustering and active contours for histopathology image segmentation and nuclei detection. In *Advanced concepts for intelligent vision systems*, pages 903–914. Springer, 2008. 77

[90] A. Hafiane, K. Palaniappan, and G. Seetharaman. UAV-Video Registration Using Block-Based Features. In *IGARSS 2008 - 2008 IEEE International Geoscience and Remote Sensing Symposium*, volume 2, pages II–1104–II–1107, 2008. 20

[91] S. Hare, A. Saffari, and P. H. Torr. Struck: Structured output tracking with kernels. *IEEE Int. Conf. on Computer Vision*, pages 263–270, 2011. 46

[92] I. Haritaoglu, D. Harwood, and L. S. Davis. W/sup 4: real-time surveillance of people and their activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):809–830, 2000. 44

[93] R. Hartley, J. Trumpf, Y. Dai, and H. Li. Rotation Averaging. *International Journal of Computer Vision*, 103(3):267–305, jan 2013. 24

[94] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 39

[95] R. I. Hartley and A. Zisserman. Multiple View Geometry in Computer Vision. 2003. 13, 15, 18, 23, 24, 64

[96] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. Exploiting the circulant structure of tracking-by-detection with kernels. *European conf. on Computer Vision*, pages 702–715, 2012. 46

[97] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):583–596, 2015. 49

[98] J. A. Hesch, D. G. Kottas, S. L. Bowman, and S. I. Roumeliotis. Camera-IMU-based localization: Observability analysis and consistency improvement. *The International Journal of Robotics Research*, 33(1):182–201, nov 2013. 59

[99] D. J. Holtkamp and A. A. Goshtasby. Precision Registration and Mosaicking of Multicamera Images. *IEEE Transactions on Geoscience and Remote Sensing*, 47:3446–3455 ST – Precision Registration and Mosaick, 2009. 20, 21

[100] B. K. Horn and B. G. Schunck. Determining optical flow. In *1981 Technical symposium east*, pages 319–331. International Society for Optics and Photonics, 1981. 40

[101] I. Huerta, M. B. Holte, T. B. Moeslund, and J. Gonzàlez. Chromatic shadow detection and tracking for moving foreground segmentation. *Image and Vision Computing*, 41:42–53, 2015. 38

[102] V. Indelman, R. Roberts, C. Beall, and F. Dellaert. Incremental light bundle adjustment. In *British Machine Vision Conference*, pages 134.1–134.11, 2012. 13, 15

[103] V. Indelman, R. Roberts, C. Beall, and F. Dellaert. Incremental light bundle adjustment. In *British Machine Vision Conference (BMVC)*, pages 134.1–134.11, 2012. 58

[104] A. Irschara, C. Hoppe, H. Bischof, and S. Kluckner. Efficient structure from motion with weak position and orientation priors. *IEEE Conf. Computer Vision and Pattern Recognition Workshops*, 2011. 13, 15, 19

[105] B. P. Jackson and a. A. Goshtasby. Adaptive Registration of Very Large Images. 2014. 20

[106] Y. Jeong, S. Member, D. Niste, and I.-s. Kweon. Pushing the envelope of modern methods for bundle adjustment. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 34(8):1605–1617, 2012. 15, 18

[107] Y. Jeong, S. Member, D. Niste, and I.-s. Kweon. Pushing the Envelope of Modern Methods for Bundle Adjustment. *IEEE Trans. in Pattern Analysis and Machine Intelligence*, 34(8):1605–1617, 2012. 58

[108] A. D. Jepson, D. J. Fleet, and T. F. El-Maraghi. Robust online appearance models for visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1296–1311, 2003. 44

[109] P.-M. Jodoin and M. Mignotte. Motion segmentation using a k-nearest-neighbor-based fusion procedure of spatial and temporal label cues. In *Image Analysis and Recognition*, pages 778–788. Springer, 2005. 38

[110] T. Jones, F. Durand, and M. Desbrun. Non-iterative, feature-preserving mesh smoothing. *ACM Trans. Graph.*, 22(3):943–949, 2003. 54, 55

[111] M. Keck, L. Galup, and C. Stauffer. Real-time tracking of low-resolution vehicles for wide-area persistent surveillance. In *2013 IEEE Workshop on Applications of Computer Vision (WACV)*, pages 441–448, 2013. 38

[112] K. Konolige. Sparse Bundle Adjustment. *Proc. of the British Machine Vision Conference*, pages 102.1–102.11, 2010. 15

[113] K. Konolige and M. Agrawal. FrameSLAM: From Bundle Adjustment to Real-Time Visual Mapping. *IEEE Transactions on Robotics*, 24(5):1066–1077, oct 2008. 58

[114] J. Kopf, M. Cohen, and R. Szeliski. First-person hyper-lapse videos. *ACM Trans. Graphics (SIGGRAPH)*, 33(4), 2014. 13, 16

[115] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. Čehovin, T. Vojir, G. Häger, A. Lukežič, and G. Fernandez. The visual object tracking vot2016 challenge results. Springer, Oct 2016. 49, 79, 81

[116] M. Kristan, J. Matas, A. Leonardis, M. Felsberg, L. Cehovin, G. Fernández, T. Vojir, G. Hager, G. Nebehay, and R. Pflugfelder. The visual object tracking vot2015 challenge results. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1–23, 2015. 49, 78, 79, 81

[117] G. Kuschk and D. Cremers. Fast and accurate large-scale stereo reconstruction using variational methods. In *ICCV Workshop on Big Data in 3D Computer Vision*, Sydney, Australia, December 2013. 57

[118] R. Lakemond, C. Fookes, and S. Sridharan. Resection-intersection bundle adjustment revisited. *ISRN Machine Vision*, 2013:1–8, 2013. 15

[119] B. Lee, K. Yun, J. Choi, and J. Y. Choi. Robust pan-tilt-zoom tracking via optimization combining motion features and appearance correlations. In *IEEE Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6, 2015. 38

[120] J. Lee, X. Cai, C.-B. Schonlieb, and D. A. Coomes. Nonparametric image registration of airborne LiDAR, hyperspectral and photographic imagery of wooded landscapes. *IEEE Transactions on Geoscience and Remote Sensing*, PP(99):1–12, 2015. 20, 21

[121] B. Leibe, E. Seemann, and B. Schiele. Pedestrian detection in crowded scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 878–885, 2005. 49

[122] M. J. Leotta, M. Dawkins, and P. Tunison. Open Source Structure-from-Motion for Aerial Video. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–9, Lake Placid, NY, USA, 2016. 8, 15, 65, 70

[123] S. Leutenegger, P. Furgale, V. Rabaud, M. Chli, K. Konolige, and R. Siegwart. Keyframe-Based Visual-Inertial SLAM Using Nonlinear Optimization. In *Proceedings of Robotics: Science and Systems*, 2013. 59

[124] M. Lhuillier. Incremental fusion of structure-from-motion and GPS using constrained bundle adjustments. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 34(12):2489–2495, 2012. 15

[125] L. Li, A. Ellis, and J. Ferryman. On fusion for robust motion segmentation. In *Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6, 2015. 38

[126] Y. Li and J. Zhu. A scale adaptive kernel correlation filter tracker with feature integration. In *European Conference on Computer Vision Workshops*, pages 254–265, 2014. 44, 49, 82

[127] J. Lim and D. Kriegman. Tracking humans using prior and learned representations of shape and appearance. In *Proceedings of IEEE International Conference on Automatic Face and Gesture Recognition*, pages 869–874, 2004. 49

[128] R. Lin, X. Cao, Y. Xu, C. Wu, and H. Qiao. Airborne moving vehicle detection for video surveillance of urban traffic. In *IEEE Intelligent Vehicles Symposium*, pages 203–208, 2009. 38

[129] Y. Lin, Q. Yu, and G. Medioni. Efficient detection and tracking of moving objects in geo-coordinates. *Machine Vision and Applications*, pages 505–520, 2010. 20

[130] M. E. Linger and a. A. Goshtasby. Aerial image registration for tracking. *IEEE Trans. Geosci. Remote Sens.*, 53(4):2137–2145, apr 2015. 20, 74

[131] Z. Liu, J. An, and Y. Jing. A simple and robust feature point matching algorithm based on restricted spatial order constraints for aerial image registration. *IEEE Transactions on Geoscience and Remote Sensing*, 50(2):514–527, 2012. 21

[132] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections, 1981. 15

[133] M. Lourakis and A. Argyros. SBA: A software package for sparse bundle adjustment. *ACM Transactions on Mathematical Software*, 36(1):30, 2009. 15, 18

[134] S. Lynen, T. Sattler, M. Bosse, J. A. Hesch, M. Pollefeys, and R. Siegwart. Get out of my lab: Large-scale, real-time visual-intertial localization. *Robotics: Science and Systems (RSS) XI*, pages 37–46, 2015. 13

[135] J. Ma, H. Zhou, J. Zhao, Y. Gao, J. Jiang, and J. Tian. Robust Feature Matching for Remote Sensing Image Registration via Locally Linear Transforming. *IEEE Transactions on Geoscience and Remote Sensing*, PP(99):1–13, 2015. 21

[136] J. McGlone, E. Mikhail, J. Bethel, and R. Mullen, editors. *Manual of Photogrammetry, Fifth Ed.* American Society of Photogrammetry and Remote Sensing, 2004. 15

[137] G. Medioni, I. Cohen, F. Bremond, S. Hongeng, and R. Nevatia. Event detection and analysis from video streams. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(8):873–889, 2001. 20

[138] X. Mei and H. Ling. Robust visual tracking and vehicle classification via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(11):2259–2272, 2011. 44

[139] A. Miropolsky and A. Fischer. Reconstruction with 3D geometric bilateral filter. In *Proceedings of the Ninth ACM Symposium on Solid Modeling and Applications*, pages 225–229, 2004. 54

[140] K. Mitra and R. Chellappa. A scalable projective bundle adjustment algorithm using the L norm. *Proceedings - 6th Indian Conference on Computer Vision, Graphics and Image Processing, ICVGIP 2008*, pages 79–86, 2008. 15

[141] E. Molina and Z. Zhu. Persistent aerial video registration and fast multi-view mosaicing. *IEEE Transactions on Image Processing*, 23(5):2184–2192, 2014. 20

[142] M. Montemerlo and S. Thrun. FastSLAM: A Scalable Method for the Simultaneous Localization and Mapping Problem in Robotics (Springer Tracts in Advanced Robotics). feb 2007. 58

[143] B. Morris and M. Trivedi. Robust classification and tracking of vehicles in traffic video streams. In *IEEE Intelligent Transportation Systems Conference*, pages 1078–1083, 2006. 38

[144] B. T. Morris and M. M. Trivedi. Learning, modeling, and classification of vehicle track patterns from live video. *IEEE Transactions on Intelligent Transportation Systems*, 9(3):425–437, 2008. 38

[145] B. T. Morris and M. M. Trivedi. Trajectory learning for activity understanding: Unsupervised, multilevel, and long-term adaptive approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(11):2287–2301, 2011. 38

[146] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Real Time Localization and 3D Reconstruction. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 363–370. IEEE, 2006. 58

[147] J. Mundy. The relationship between photogrammetry and computer vision J.L. Mundy GE Corporate Research and Development Schenectady, NY 12309. *In Integrating Photogrammetric Techniques With Scene Analysis and Machine Vision, SPIE*, 1944, 1993. 15

[148] J. Munkres. Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 5(1):32–38, 1957. 51, 52

[149] S. Nath and K. Palaniappan. Adaptive robust structure tensors for orientation estimation and image segmentation. *Lecture Notes in Computer Science (ISVC)*, 3804:445–453, 2005. 19

[150] T. Nawaz, J. Boyle, L. Li, and J. Ferryman. Tracking performance evaluation on pets 2015 challenge datasets. In *IEEE Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6, 2015. 38

[151] S. Niko and P. Protzel. Towards using sparse bundle adjustment for robust stereo odometry in outdoor terrain. In *TARO*, volume 2, pages 206–213, 2006. 18

[152] D. Nistér. An efficient solution to the five-point relative pose problem. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 26(6):756–770, 2004. 13, 65

[153] K. Nummiaro, E. Koller-Meier, and L. Van Gool. An adaptive color-based particle filter. *Image and Vision Computing*, 21(1):99–110, 2003. 46

[154] G. S. Palaniappan, G. Gasperas, and K. A piecewise affine model for image registration in nonrigid motion analysis. In *Image Processing, 2000. Proceedings. 2000 International Conference on*, pages 561–564, 2000. 20

[155] K. Palaniappan, F. Bunyak, P. Kumar, I. Ersoy, S. Jaeger, K. Ganguli, A. Haridas, J. Fraser, R. Rao, and G. Seetharaman. Efficient feature extraction and likelihood fusion for vehicle tracking in low frame rate airborne video. In *13th International Conference on Information Fusion*, 2010. 13

[156] K. Palaniappan, F. Bunyak, P. Kumar, I. Ersoy, S. Jaeger, K. Ganguli, A. Haridas, J. Fraser, R. M. Rao, and G. Seetharaman. Efficient feature extraction and likelihood fusion for vehicle tracking in low frame rate airborne video. In *International Conferenceon on Information Fusion*, pages 1–8, 2010. 43, 44, 47

[157] K. Palaniappan, I. Ersoy, and S. K. Nath. Moving object segmentation using the flux tensor for biological video microscopy. *Lecture Notes in Computer Science (PCM)*, 4810:483–493, 2007. 40

[158] K. Palaniappan, M. Poostchi, H. Aliakbarpour, R. Viguier, J. Fraser, F. Bunyak, A. Basharat, S. Suddarth, E. Blasch, R. M. Rao, et al. Moving object detection for vehicle tracking in wide area motion imagery using 4d filtering. In *IEE International Conference on Pattern Recognition*, pages 2830–2835, 2016. 44

[159] K. Palaniappan, M. Poostchi, H. Aliakbarpour, R. Viguier, J. Fraser, F. Bunyak, A. Basharat, S. Suddarth, E. Blash, R. Rao, and G. Seetharaman. Moving object detection for vehicle tracking in wide area motion imagery using 4D filtering. In *IEEE International Conference on Pattern Recognition (ICPR)*, 2016. 4, 13, 14, 16

[160] K. Palaniappan, R. Rao, and G. Seetharaman. Wide-area persistent airborne video: Architecture and challenges. In *Distributed Video Sensor Networks: Research Challenges and Future Directions*, pages 349–371. Springer, 2011. 13, 16, 38

[161] F. Panoff-Eliet, B. Berlin, and P. Kay. Basic color terms. their universality and evolution, 1971. 54

[162] N. Paragios and R. Deriche. Geodesic active contours and level sets for the detection and tracking of moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(3):266–280, 2000. 38

[163] D. H. Parks and S. S. Fels. Evaluation of background subtraction algorithms with post-processing. In *IEEE Advanced Video and Signal Based Surveillance*, pages 192–199, 2008. 38

[164] H. A. Patel and D. G. Thakore. Moving object tracking using kalman filter. *International Journal of Computer Science and Mobile Computing (IJCSMC)*, 2(4):326–332, 2013. 38

[165] R. Pelapur, F. Bunyak, G. Seetharaman, and K. Palaniappan. Vehicle detection and orientation estimation using the radon transform. In *Proc. SPIE Conf. Geospatial InfoFusion III (Defense, Security and Sensing: Sensor Data and Information Exploitation)*, volume 8747, page 87470I, 2013. 38

[166] R. Pelapur, S. Candemir, F. Bunyak, M. Poostchi, G. Seetharaman, and K. Palaniappan. Persistent target tracking using likelihood fusion in wide-area and full motion video sequences. In *15th Int. Conf. Information Fusion*, pages 2420–2427, 2012. 13, 38

[167] R. Pelapur, S. Candemir, F. Bunyak, M. Poostchi, G. Seetharaman, and K. Palaniappan. Persistent target tracking using likelihood fusion in wide-area and full motion video sequences. In *International Conferenceon on Information Fusion*, pages 2420–2427, 2012. 43, 49

[168] R. Pelapur, S. Candemir, F. Bunyak, M. Poostchi, G. Seetharaman, and K. Palaniappan. Persistent target tracking using likelihood fusion in wide-area and full motion video sequences. *Int. Conf. on Information Fusion*, pages 2420–2427, 2012. 44

[169] R. Pelapur, K. Palaniappan, and G. Seetharaman. Robust orientation and appearance adaptation for wide-area large format video object tracking. In *9th IEEE Int. Conf. Advanced Video and Signal-Based Surveillance*, 2012. 13, 38

[170] R. Pelapur, K. Palaniappan, and G. Seetharaman. Robust orientation and appearance adaptation for wide-area large format video object tracking. *IEEE Int. Conf. on Advanced Video and Signal-Based Surveillance*, pages 337–342, 2012. 44

[171] O. Pele and M. Werman. Improving perceptual color difference using basic color terms. *arXiv preprint arXiv:1211.5556*, 2012. 54

[172] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. *European Conf. on Computer Vision*, pages 661–675, 2002. 46

[173] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1201–1208, 2011. 82, 83

[174] M. Pollefeys, D. Nistér, J.-M. Frahm, A. Akbarzadeh, et al. Detailed real-time urban 3D reconstruction from video. *Int. Journal of Computer Vision*, 78(2-3):143–167, oct 2007. 13, 15

[175] M. Poostchi, H. Aliakbarpour, R. Viguier, F. Bunyak, K. Palaniappan, and D. C. Washington. Semantic Depth Map Fusion for Moving Vehicle Detection in Aerial Video. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 32–40, 2016. 4, 13, 14, 16

[176] M. Poostchi, F. Bunyak, and K. Palaniappan. Feature selection for appearance-based vehicle tracking in geospatial video. In *Proc. SPIE Conf. Geospatial InfoFusion III (Defense, Security and Sensing: Sensor Data and Information Exploitation)*, volume 8747, page 87470G, 2013. 38

[177] F. Porikli, F. Brémond, S. L. Dockstader, J. Ferryman, A. Hoogs, B. C. Lovell, S. Pankanti, B. Rinner, P. Tu, P. L. Venetianer, et al. Video surveillance: past, present, and now the future. *IEEE Signal Processing Magazine*, 30(3):190–198, 2013. 38

[178] H. Possegger, T. Mauthner, and H. Bischof. In defense of color-based model-free tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2113–2120, 2015. 44

[179] A. Prioletti, A. Mogelmose, P. Grisleri, M. M. Trivedi, A. Broggi, and T. B. Moeslund. Part-based pedestrian detection and feature-based tracking for driver assistance: real-time, robust algorithms, and evaluation. *IEEE Transactions on Intelligent Transportation Systems*, 14(3):1346–1359, 2013. 38

[180] M. D. Pritt. Fast Orthorectified Mosaics of Thousands of Aerial Photographs from Small UAVs. *Applied Imagery Pattern Recognition Workshop (AIPR), IEEE*, 2014. 20

[181] K. a. Redmill, J. I. Martin, and U. Ozguner. Aerial image registration incorporating GPS/IMU data. *Proceedings of SPIE*, 7347(1):73470H–73470H–15, 2009. 21

[182] Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International journal of computer vision*, 40(2):99–121, 2000. 54

[183] I. Saleemi and M. Shah. Multiframe Many-Many Point Correspondence for Vehicle Tracking in High Density Wide Area Aerial Videos. *International Journal of Computer Vision*, 104(2):198–219, 2013. 20

[184] J. Schönberger, F. Fraundorfer, and J.-M. Frahm. Structure-from-Motion for MAV image sequence analysis with photogrammetric applications. In *International Archives of the Photogrammetry, Remote Sensing, and Spatial Information Sciences*, volume XL-3, pages 305–312, 2014. 10, 13, 15, 16, 60, 65

[185] F. Schubert and K. Mikolajczyk. Robust registration and filtering for moving object detection in aerial videos. In *Proceedings of the 2014 22Nd International Conference on Pattern Recognition*, ICPR '14, pages 2808–2813, Washington, DC, USA, 2014. 38

[186] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 519–528, 2006. 60, 70

[187] G. Sharma, W. Wu, and E. N. Dalal. The ciede2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations. *Color Research & Application*, 30(1):21–30, 2005. 54

[188] K. I. J. Shawn Recker, Christiaan Gribble, Mikhail Shashkov, Mario Yepez, Mauricio Hess-Flores. Depth Data Assisted Structure-from-Motion Parameter Optimization and Feature Track Correction. In *Applied Imagery Pattern Recognition Workshop (AIPR)*, 2014. 15

[189] J. Shin, S. Kim, S. Kang, S.-W. Lee, J. Paik, B. Abidi, and M. Abidi. Optical flow-based real-time object tracking using non-prior training active feature model. *Real-Time Imaging*, 11(3):204–218, 2005. 44

[190] M. S. Shirazi and B. Morris. Vision-based vehicle queue analysis at junctions. In *Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6, 2015. 38

[191] G. Sibley, C. Mei, I. Reid, and P. Newman. Adaptive relative bundle adjustment. In *Proc. Robots: Science and Systems*, 2009. 58

[192] S. Sivaraman and M. M. Trivedi. Integrated lane and vehicle detection, localization, and tracking: A synergistic approach. *IEEE Transactions on Intelligent Transportation Systems*, 14(2):906–917, 2013. 38

[193] S. Sivaraman and M. M. Trivedi. Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis. *IEEE Transactions on Intelligent Transportation Systems*, 14(4):1773–1795, 2013. 38

[194] S. Sivaraman and M. M. Trivedi. Active learning for on-road vehicle detection: A comparative study. *Machine vision and applications*, 25(3):599–611, 2014. 38

[195] K. Smith, D. Gatica-Perez, and J.-M. Odobez. Using particles to track varying numbers of interacting people. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 962–969, 2005. 49

[196] N. Snavely, S. M. Seitz, and R. Szeliski. Modeling the world from Internet photo collections. *International Journal of Computer Vision*, 80:189–210, 2008. 20

[197] H. Stewenius, C. Engels, and D. Nistér. Recent developments on direct relative orientation. *ISPRS Journal of Photogrammetry and Remote Sensing*, 60(4):284–294, 2006. 15

[198] H. S. Stone, M. T. Orchard, E. C. Chang, and S. a. Martucci. A fast direct Fourier-based algorithm for subpixel registration of images. *IEEE Transactions on Geoscience and Remote Sensing*, 39(10):2235–2243, 2001. 20

[199] C. Strecha, W. von Hansen, L. V. Gool, P. Fua, and U. Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. *IEEE Conf. on Computer Vision and Pattern Recognition*, 2008. 60, 70

[200] Z. H. Sun, M. Leotta, A. Hoogs, R. Blue, R. Neuroth, J. Vasquez, A. Perera, M. Turek, and E. Blasch. Vehicle change detection from aerial imagery using detection response maps. In *SPIE Defense+ Security*, pages 908906–908906. International Society for Optics and Photonics, 2014. 38

[201] C. N. Taylor. Improved evaluation of geo-registration algorithms for airborne EO/IR imagery. In *SPIE, Geospatial Infofusion III*, volume 8747, page 874709, 2013. 20

[202] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *IEEE International Conference on Computer Vision*, pages 59–66, 1998. 54

[203] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment – a modern synthesis. In *Vision Algorithms: Theory and Practice (W. Triggs, A. Zisserman, and R. Szeliski Eds.)*, pages 298–372, 2000. 15, 18, 19

[204] B. Triggs, P. F. Mclauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle Adjustment — A Modern Synthesis. In *Lecture Notes in Computer Science (Vision Algorithms)*, volume 1883, pages 298–372. Springer Berlin Heidelberg, 2000. 58

[205] D. Turner, A. Lucieer, and L. Wallace. Direct Georeferencing of Ultrahigh-Resolution UAV Imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 52(5):2738–2745, may 2014. 20

[206] J. Van De Weijer, C. Schmid, J. Verbeek, and D. Larlus. Learning color names for real-world applications. *IEEE Trans. on Image Processing*, 18(7):1512–1523, 2009. 44, 46, 54

[207] E. Vasquez, Juan and Hytla, Patrick and Asari, Vijayan and Jackovitz, Kevin and Balster. Registration of Region of Interest for Object Tracking Applications in Wide Area Motion Imagery. In *the IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, pages 1–8, 2012. 20

[208] R. Viguier, C. C. Lin, H. AliAkbarpour, F. Bunyak, S. Pankanti, G. Seetharaman, and K. Palaniappan. Automatic Video Content Summarization Using Geospatial Mosaics of Aerial Imagery. *2015 IEEE International Symposium on Multimedia (ISM)*, pages 249–253, 2015. 4, 13, 16

[209] R. Viguier, C. C. Lin, K. Swaminathan, A. Vega, A. Buyuktosunoglu, S. Pankanti, P. Bose, H. Akbarpour, F. Bunyak, K. Palaniappan, and G. Seetharaman. Resilient mobile cognition: Algorithms, innovations, and architectures. *Proceedings of the 33rd IEEE International Conference on Computer Design, ICCD 2015*, pages 728–731, 2015. 4, 13, 16

[210] R. Wang, F. Bunyak, G. Seetharaman, and K. Palaniappan. Static and moving object detection using flux tensor with split gaussian models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 414–418, 2014. 38

[211] L. Wen, D. Du, Z. Cai, Z. Lei, M. Chang, H. Qi, J. Lim, M. Yang, and S. Lyu. UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking. *arXiv CoRR*, abs/1511.04136, 2015. 10, 54, 81, 82, 83

[212] L. Wen, W. Li, J. Yan, Z. Lei, D. Yi, and S. Z. Li. Multiple target tracking based on undirected hierarchical relation hypergraph. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1282–1289, 2014. 82, 83

[213] C. R. Wren, A. Azarbayejani, T. Darrell, and A. P. Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):780–785, 1997. 38

[214] C. Wu. SiftGPU: A GPU implementation of Scale Invaraint Feature Transform (SIFT), 2007. 60

[215] C. Wu. Towards linear-time incremental structure from motion. *International Conference on 3D Vision*, pages 127–134, 2013. 15, 65

[216] C. Wu. Critical configurations for radial distortion self-calibration. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 25–32, 2014. 18

[217] C. Wu, S. Agarwal, B. Curless, and S. M. Seitz. Multicore bundle adjustment. In *IEEE Conf. Computer Vision Pattern Recognition*, pages 3057–3064, 2011. 10, 15, 60, 65

[218] C. Wu, S. Agarwal, B. Curless, and S. M. Seitz. Multicore bundle adjustment. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3057–3064, 2011. 39

[219] J. Xiao, H. Cheng, H. Sawhney, and F. Han. Vehicle detection and tracking in wide field-of-view aerial video. In *Computer Vision and Pattern Recognition (CVPR)*, pages 679–684, 2010. 38

[220] J. Xiao, R. Stolkin, and A. Leonardis. Single target tracking using adaptive clustered decision trees and dynamic multi-level appearance models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4978–4987, 2015. 44

[221] K. Zhang, L. Zhang, and M.-H. Yang. Real-time compressive tracking. *European Conf. on Computer Vision*, pages 864–877, 2012. 46

[222] Z. Zhang. Camera Calibration. *Emerging Topics in Computer Vision*, pages 4–43, 2004. 18

[223] Z. Zhang and Y. Shan. Incremental Motion Estimation Through Local Bundle Adjustment. (May), 2001. 15

[224] Z. Zhu, A. R. Hanson, and E. M. Riseman. Generalized Parallel-Perspective Stereo Mosaics from Airborne Video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):226–237, 2004. 20

[225] Z. Zivkovic and B. Krose. An em-like algorithm for color-histogram-based object tracking. *IEEE Conf. on Computer Vision and Pattern Recognition*, 1:I–798, 2004. 46